K. Srinivas
C.A.J. Fletcher

Computational Techniques for Fluid Dynamics

Scientific Computation

A Solutions Manual

Springer

Springer Series in Computational Physics

K. Srinivas
C. A. J. Fletcher

Computational Techniques for Fluid Dynamics

A Solutions Manual

Springer-Verlag

K. Srinivas C. A. J. Fletcher

Computational Techniques for Fluid Dynamics

A Solutions Manual

With 53 Figures

Springer-Verlag
Berlin Heidelberg New York
London Paris Tokyo
Hong Kong Barcelona
Budapest

Dr. Karkenahalli Srinivas
Department of Aeronautical Engineering
The University of Sydney
New South Wales 2006, Australia

Dr. Clive A. J. Fletcher
Department of Mechanical Engineering
The University of Sydney
New South Wales 2006, Australia

ISBN 3-540-54304-X Springer-Verlag Berlin Heidelberg New York
ISBN 0-387-54304-X Springer-Verlag New York Berlin Heidelberg

Typesetting: Camera ready by authors
55/3140 – 5 4 3 2 1 0 – Printed on acid-free paper

Preface

This complementary text provides detailed solutions for the problems that appear in Chapters 2 to 18 of *Computational Techniques for Fluid Dynamics* (CTFD), Second Edition. Consequently there is no Chapter 1 in this solutions manual. The solutions are indicated in enough detail for the serious reader to have little difficulty in completing any intermediate steps.

Many of the problems require the reader to write a computer program to obtain the solution. Tabulated data, from computer output, are included where appropriate and coding enhancements to the programs provided in CTFD are indicated in the solutions. In some instances completely new programs have been written and the listing forms part of the solution. All of the program modifications, new programs and input/output files are available on an IBM-compatible floppy direct from C.A.J. Fletcher.

Many of the problems are substantial enough to be considered mini-projects and the discussion is aimed as much at encouraging the reader to explore extensions and what-if scenarios leading to further development as at providing neatly packaged solutions. Indeed, in order to give the reader a better introduction to CFD reality, not all the problems do have a "happy ending". Some suggested extensions fail; but the reasons for the failure are illuminating.

Although the solutions manual is targeted at course instructors it is recognised that the manual will be of considerable direct value to the incipient computational fluid dynamicist. However, where the manual is being used without formal instruction it is strongly recommended that the relevant sections of CTFD be read very thoroughly, the problem be answered fully in writing and then, *and only then*, the solutions manual be consulted.

In preparing this solutions manual we have been greatly assisted by Susan Gonzales in preparing the text, by Dr. D.J. Auld and Mr. A. Tan and many other colleagues and students in checking the solutions and by Professor W. Beiglböck and his colleagues at Springer-Verlag. We are very grateful to all of them. Finally we acknowledge the substantial debt we owe to Usha Srinivas, Gautham Srinivas and Mary Fletcher for their unwavering support and encouragement.

Sydney,
November 1991

K. Srinivas
C.A.J. Fletcher

Contents

The chapter headings listed below correspond to the main text: *Computational Techniques for Fluid Dynamics I/II*. Only those chapters containing problems are given; the page numbers are those on which the solutions to the problems can be found in the manual.

CTFD Solutions Manual: Chapter 2

Partial Differential Equations

2.1 **a)** We can interpret Laplace's equation as (2.8) with $A = 1$, $B = 0$, $C = 1$, $H = 0$. Therefore (2.8) is transformed into (2.17) with

$$\begin{aligned} A' &= \xi_x^2 + \xi_y^2 \\ B' &= 2\xi_x\eta_x + 2\xi_y\eta_y \\ C' &= \eta_x^2 + \eta_y^2, \end{aligned}$$

and H' involves $\partial\varphi/\partial\xi$ and $\partial\varphi/\partial\eta$.

The character of (2.17) is determined by $(B')^2 - 4A'C'$. For the present example,

$$(B')^2 - 4A'C' = -4(\xi_x\eta_y - \xi_y\eta_x)^2 = -4J^2.$$

Since this is always negative Laplace's equation is elliptic.

b) The procedure is as above. For the wave equation, $A = 1$, $B = 0$, $C = -1$, $H = 0$ in (2.8). Therefore in (2.17),

$$\begin{aligned} A' &= \xi_x^2 - \xi_y^2 \\ B' &= 2\xi_x\eta_x - 2\xi_y\eta_y \\ C' &= \eta_x^2 - \eta_y^2, \end{aligned}$$

and $(B')^2 - 4A'C' = 4J^2$.

Since this is always positive the wave equation is hyperbolic.

2.2 Let $\partial u/\partial x = p$ and $\partial^2 u/\partial x^2 = \partial p/\partial x = q$.

Therefore we replace $\partial u/\partial t + u\partial u/\partial x + \partial^3 u/\partial x^3 = 0$ with

$$\partial u/\partial t + up + \partial q/\partial x = 0$$

$$\partial u/\partial x - p = 0$$

$$\partial p/\partial x - q = 0.$$

Applying the Fourier method (Sect. 2.1.5) gives

$$\begin{bmatrix} i\lambda_t & u & i\lambda_x \\ i\lambda_x & -1 & 0 \\ 0 & i\lambda_x & 1 \end{bmatrix} \begin{bmatrix} \hat{u} \\ \hat{p} \\ \hat{q} \end{bmatrix} = 0,$$

or $i\{\lambda_t + u\lambda_x - \lambda_x^3\} = 0$.

Clearly the same characteristic (symbolic) polynomial would have been obtained by substituting directly into the original governing equation.

The solution is $\lambda_t = -u\lambda_x - \lambda_x^3$.

Since one real solution is obtained we deduce that the equation is parabolic. However since the solution is a wave without damping, (9.25) and (9.27), the governing equation is better described as being 'hyperbolic' in character.

2.3 Using the technique described in Sect. 2.1.4,

$$\underline{\mathbf{A}} = \begin{bmatrix} 1 & 0 & 0 \\ u & 0 & 1 \\ 0 & u & 0 \end{bmatrix}, \quad \underline{\mathbf{B}} = \begin{bmatrix} 0 & 1 & 0 \\ v & 0 & 0 \\ 0 & v & 1 \end{bmatrix},$$

and $\det[\underline{\mathbf{A}}(dy/dx) - \underline{\mathbf{B}}] = 0$ gives

$$-(dy/dx)^2\{udy/dx - v\} - \{udy/dx - v\} = 0$$

or $dy/dx = v/u$ *and* $\pm\, i$.

Thus two roots, associated with the pressure/continuity interaction, indicate elliptic behaviour and one root, associated with the convective operator, indicates hyperbolic behaviour. Thus the system is of mixed type with the elliptic behaviour having the dominant influence. Symbolic analysis (Sect. 2.1.5) produces the same result.

2.4 Interpreting $\partial^2 u/\partial x \partial t = 0$ as (2.1) implies $A = 0$, $B = 1$, $C = 0$. Since $B^2 - 4AC > 0$, this equation is hyperbolic.

The system

$$\partial u/\partial t - v = 0$$

$$\partial v/\partial x = 0,$$

can be combined as $\partial(\partial u/\partial t - v)/\partial x = 0$,

or $\partial^2 u/\partial x\partial t - \partial v/\partial x = 0$,

or $\partial^2 u/\partial x\partial t = 0$, as before.

Thus the system is hyperbolic. The system can also be written

$$\partial u/\partial t + \alpha\partial u/\partial x = v, \tag{1}$$

and

$$\beta\partial v/\partial t + \partial v/\partial x = 0, \tag{2}$$

where α, $\beta = 0$.

Equations (1) and (2) have characteristic directions, $dx/dt = \alpha$ and $1/\beta$ respectively. Since α, $\beta = 0$, the x and t axes are characteristics.

2.5 We can interpret

$$\partial^2 u/\partial t^2 - \beta\partial^2 u/\partial x^2 + u = 0, \tag{2.94}$$

as a special case of (2.1) with $A = 1$, $B = 0$, $C = -\beta$. Since $B^2 - 4AC = 4\beta > 0$ if β is positive, (2.94) is hyperbolic. The characteristic directions are given by

$$(dx/dt)^2 - \beta = 0 \quad \text{or} \quad dx/dt = \pm\beta^{\frac{1}{2}}.$$

If we set $v = \partial u/\partial x$ and $w = \partial u/\partial t$ the equivalent system is obtained,

$$\begin{aligned} &\partial u/\partial t - w = 0 \\ &\partial v/\partial t - \partial w/\partial x = 0 \\ &\partial w/\partial t - \beta\partial v/\partial x + u = 0. \end{aligned} \tag{2.95}$$

Applying the procedure of Sect. 2.1.4 leads to

$$\underline{\mathbf{A}}(dx/dt) - \underline{\mathbf{B}} \equiv \begin{bmatrix} dx/dt & 0 & 0 \\ 0 & dx/dt & 1 \\ 0 & \beta & dx/dt \end{bmatrix}.$$

Consequently (2.29) gives $dx/dt[(dx/dt)^2 - \beta] = 0$ which has the solution, $dx/dt = 0, \pm\beta^{\frac{1}{2}}$. Since all roots are real the system, (2.95), is hyperbolic and the extra characteristic, $dx/dt = 0$, arises from the definition of w, which uncouples from the other two equations in determining the character of the equation system.

2.6 From $p = k\rho^\gamma$ and $a^2 = \partial p/\partial\rho = \gamma p/\rho$

$$d\rho = \{2\rho/(\gamma-1)a\}\, da \quad \text{and} \quad dp = \{2\rho a/(\gamma-1)\}\, da.$$

Therefore the governing equations for one-dimensional, unsteady isentropic inviscid compressible flow can be written,

$$\partial u/\partial t + u\partial u/\partial x + 2a/\ (\gamma-1)\ \partial a/\partial x = 0$$

$$\partial a/\partial t + u\partial a/\partial x + (\gamma-1)a/2\ \partial u/\partial x = 0.$$

Applying the method of Sect. 2.1.4

$$\underline{\mathbf{A}}(dx/dt) - \underline{\mathbf{B}} \equiv \begin{bmatrix} dx/dt - u & 2a/(\gamma-1) \\ (\gamma-1)a/2 & dx/dt - u \end{bmatrix},$$

and $\det[\underline{\mathbf{A}}(dx/dt) - \underline{\mathbf{B}}] = 0$ gives $(dx/dt - u)^2 = a^2$

or $dx/dt = u \pm a$,

and since both characteristics are real, the system is hyperbolic.

2.7 a) We replace $\partial\varphi/\partial t - \alpha\partial^2\varphi/\partial x^2 = 0$

with $\alpha\partial p/\partial x - \partial\varphi/\partial t = 0$

$\partial\varphi/\partial x = p.$

Using the procedure of Sect. 2.1.4,

$$\det[\underline{\mathbf{A}}(dt/dx) - \underline{\mathbf{B}}] = 0 \quad \text{gives} \quad \alpha(dt/dx)^2 = 0.$$

The single characteristic direction, $dt/dx = 0$, indicates that the equation is parabolic.

b) By considering a surface $y = constant$ the character of the governing equation is the same as in 2.7(a), *i.e.* it is parabolic. In a similar way on a surface $x = constant$ the governing equation is parabolic in t. However on a surface $t = constant$ the governing equation is elliptic.

2.8 We can interpret

$$\partial u/\partial t + 2c\partial u/\partial x - d\partial^2 u/\partial x^2 = 0,$$

as (2.1) with $A = 0$, $B = 0$ and $C = -d$, which indicates that $B^2 - 4AC = 0$, i.e. the equation is parabolic.

A separable solution is sought of the form

$$u = \exp(cx/d)f(t).$$

This satisfies the initial condition if $f(0) = 1.0$. Also a separable solution of the form $g(x)\exp(-c^2t/d)$ satisfies the boundary conditions if $g(0) = 1.0$. Therefore the solution $u = \exp(cx/d)\exp(-c^2t/d)$ satisfies both the initial and boundary conditions. For this solution,

$$\partial u/\partial t = -c^2u/d, \quad \partial u/\partial x = cu/d, \quad \partial^2 u/\partial x^2 = c^2u/d^2.$$

Substitution indicates that the governing equation is identically satisfied. Thus the required solution is

$$u = \exp(cx/d)\exp(-c^2t/d).$$

2.9 Applying the Fourier method (Sect. 2.1.5) leads to

$$\begin{bmatrix} i\sigma_x & i\sigma_y \\ iu\sigma_x + v\sigma_y + \frac{1}{Re}\sigma_y^2 & 0 \end{bmatrix} \begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix} = 0,$$

and det[ ] = 0 gives $\sigma_y[u\sigma_x + v\sigma_y - \frac{1}{Re}\sigma_y^2] = 0.$

Considering only the principal part, $\sigma_x = i\sigma_y^2/uRe.$

Thus the character is determined solely by the x-momentum equation and this is parabolic in the x direction if u is positive. Once the u solution is obtained at any downstream station, the continuity equation provides an ordinary differential equation for v.

Suitable boundary conditions are $u = v = 0$ at $y = 0$ and $u = u_e$ at $y = \delta$. Suitable initial (upstream) conditions are $u = u_o(y)$ at $x = x_o$. Given $u_o(y)$ the corresponding initial v profile, $v_o(y)$, is obtained by solving

$$-u_o\partial v/\partial y + v\partial u_o/\partial y - (1/Re)\partial^2 u_o/\partial y^2 = 0.$$

2.10 a) Applying the Fourier method, Sect. (2.1.5), directly to

$$\partial u/\partial x + \partial v/\partial y = 0 \tag{2.96a}$$

and

$$\partial u/\partial y - \partial v/\partial x = 0, \tag{2.96b}$$

gives det [ ] $= \lambda_x^2 + \lambda_y^2 = 0$ or $\lambda_x/\lambda_y = \pm i$,

i.e. two imaginary roots occur indicating that the system is elliptic.

b) If $u = \partial\varphi/\partial x$, $v = \partial\varphi/\partial y$, (2.96a) becomes

$$\partial^2\varphi/\partial x^2 + \partial^2\varphi/\partial y^2 = 0, \tag{1}$$

and (2.96b) is satisfied identically. Interpreting (1) as (2.1) indicates that $A = 1$, $B = 0$ and $C = 1$, so that $B^2 - 4AC < 0$ and the equation is elliptic.

2.11 If $u = x/(x^2 + y^2)$, $v = y/(x^2 + y^2)$,

then $\partial u/\partial x = -(x^2 - y^2)/(x^2 + y^2)^2$; $\partial u/\partial y = -2xy/(x^2 + y^2)^2$

and $\partial v/\partial x = -2xy/(x^2 + y^2)^2$; $\partial v/\partial y = (x^2 - y^2)/(x^2 + y^2)^2$.

So that, by direct substitution, (2.96a) and (2.96b) are satisfied.

2.12 Applying the Fourier method, Sect. (2.1.5), gives the symbolic polynomial (α= 1/Re)

$$[iu\lambda_x + iv\lambda_y + \alpha(\lambda_x^2 + \lambda_y^2)]^2 = 0.$$

Retaining only the principal part gives

$$\alpha(\lambda_x^2 + \lambda_y^2)^2 = 0,$$

which indicates that each equation provides two imaginary roots and is separately elliptic.

The choice of D is such that, with $D_x \equiv \partial D/\partial x$ etc.,

$$u = -(2/Re)(D_x/D) \quad \text{and} \quad v = -(2/Re)(D_y/D),$$
$$u_x = -(2/Re)(D_{xx}/D) + 0.5Re\ u^2,$$
$$u_{xx} = -(2/Re)(D_{xxx}/D) - u(D_{xx}/D) + Re\ uu_x,$$

so that

$$uu_x - (1/Re)u_{xx} = (2/Re^2)(D_{xxx}/D) + (u/Re)(D_{xx}/D). \tag{1}$$

In a similar way,

$$vu_y-(1/Re)u_{yy} = (2/Re^2)(D_{xyy}/D)+(v/Re)(D_{xy}/D)+0.5(vu_y-uv_y).(2)$$

Combining (1) and (2) gives,

$$uu_x + vu_y - (1/Re)(u_{xx}+u_{yy}) =(2/Re^2D)[(D_{xx}+D_{yy})_x - (D_x/D)(D_{xx}+D_{yy})].$$

In a similar way,

$$uv_x + vv_y - (1/Re)(v_{xx}+v_{yy}) =(2/Re^2D)[(D_{xx}+D_{yy})_y - (D_y/D)(D_{xx}+D_{yy})].$$

But $D_{xx}+D_{yy}=0$ for the particular choice of D. Therefore both equations are satisfied.

2.13 To solve $\partial^2T/\partial x^2+\partial^2T/\partial y^2=0$ for $0\le x\le 1,\ 0\le y\le 1,$

we assume a separation-of-variables solution,

$$T(x,y)=R(x)S(y),$$

so that the governing equation is equivalent to

$$-(1/R)\partial^2R/\partial x^2=(1/S)\partial^2S/\partial y^2=c.$$

Let $c=k^2\pi^2$ so that the general solution, T, is

$$T=\sum_{k=1}^{\infty}[A'_k\sin(k\pi x)+B_k\cos(k\pi x)][C_k\exp(k\pi y)+D_k\exp(-k\pi y)].$$

Satisfying the boundary conditions, $T(0,y)=T(1,y)=0$ gives $B_k=0$.

Satisfying the boundary condition, $T(x,1)=0$ gives

$$C_k=-D_k\ \exp(-k\pi)/\exp(k\pi).$$

Thus the general solution can be written

$$T=\sum_{k=1}^{\infty}A_k\sin(k\pi x)\sinh[k\pi(y-1)].$$

But

$$T(x,0)=T_o=-\sum_{k=1}^{\infty}A_k\sin(k\pi x)\sinh[k\pi]$$

so that $A_k=(2T_o/k\pi)\,[(-1)^k-1]/[\sinh(k\pi)]$.

2.14 To solve $\partial\varphi/\partial t-\alpha\partial^2\varphi/\partial x^2=0$ for $t\ge 0$ and $0\le x\le 1,$

we assume a separation-of-variables solution,

$$\varphi(t,x)=P(t)R(x),$$

so that the governing equation is equivalent to

$$(1/P)\partial P/\partial t=(\alpha/R)\partial^2R/\partial x^2=-\alpha(k\pi)^2.$$

A general solution can be written as

$$\varphi=\varphi_Rx+\sum_{k=1}^{\infty}[A'_k\exp(-\alpha k^2\pi^2t)+B_k\exp(\alpha k^2\pi^2t)][C_k\sin(k\pi x) + D_k\cos(k\pi x)].$$

Satisfying the boundary conditions, $\varphi=0$ at $x=0$, gives $D_k=0$. Then the boundary condition, $\varphi=\varphi_R$ at $x=1$ is also satisfied. For the solution to be bounded at large t, $B_k=0$.

At $t=0$,

$$\varphi=\varphi_Rx+\sum_{k=1}^{\infty}A_k\ \sin(k\pi x)=0,$$

and A_k is obtained from

$$0=\varphi_R\int_0^1 x\sin(k\pi x)\ dx+A_k\int_0^1\sin^2(k\pi x)\ dx,$$

or $A_k=(-1)^k\ 2\varphi_R/(k\pi)$, so that the solution is

$$\varphi=\varphi_Rx+\sum_{k=1}^{\infty}[2\varphi_R\ (-1)^k\ \exp(-\alpha k^2\pi^2t)\sin(k\pi x)]/k\pi.$$

2.15 Equation (2.75) is $\partial u/\partial t-\partial^2u/\partial x^2=0$.

For fixed y we assume a solution of the form

$$u=\{A/(4\pi t)^{1/2}\}\exp\{-(x-y)^2/4t\}\ .$$

Thus

$$\partial u/\partial t=-0.5\{4\pi u/(4\pi t)+(x-y)^2u/(4t^2)\}$$
$$=\{-0.5/t+(x-y)^2/(4t^2)\}\,u,$$

$$\partial u/\partial x=-\{2(x-y)/4t\}u\ ,\ \text{and}$$

$$\partial^2u/\partial x^2=-u/2t-\{(x-y)/2t\}\partial u/\partial x=\{-1/2t+(x-y)^2/4t^2\}\,u.$$

Clearly $\partial u/\partial t-\partial^2u/\partial x^2=0$ as required.

CTFD Solutions Manual: Chapter 3

Preliminary Computational Techniques

3.1 The second derivative, d^2T/dx^2, is represented by the one-sided formula,

$$d^2T/dx^2 \approx aT_j + bT_{j+1} + cT_{j+2} + dT_{j+3}\ .$$

Expanding the RHS terms as Taylor series about the point, j, one gets,

$$d^2T/dx^2 \approx (a+b+c+d)T + (b+2c+3d)T_x\Delta x + (b+4c+9d)\frac{\Delta x^2}{2}T_{xx}$$
$$+ (b+8c+27d)\frac{\Delta x^3}{6}T_{x^3} + (b+16c+81d)\frac{\Delta x^4}{24}T_{x^4} + O(H)\ .$$

Equating the coefficients of the successive derivatives in the Taylor series expansion until enough equations are available to solve for a, b, c and d, gives

$$a+b+c+d=0;\quad b+2c+3d=0;\quad b+4c+9d=2/\Delta x^2 \ \text{and}\ b+8c+27d=0.$$

Solving for a, b, c and d, we have

$$a = 2/\Delta x^2\ ,\ b=-5/\Delta x^2\ ,\ c = 4/\Delta x^2 \ \text{and}\ \ d=-1/\Delta x^2\ .$$

The leading term in the truncation error is $-(11/12)\Delta x^2 T_{x^4}$.

3.2 a) The given scheme is

$$(0.5T_j^{n-1} - 2T_j^n + 1.5T_j^{n+1})/\Delta t - \alpha(1+d)(T_{j-1} - 2T_j + T_{j+1})^n/\Delta x^2$$
$$+\ \alpha d(T_{j-1} - 2T_j + T_{j+1})^{n-1}/\Delta x^2 = 0\ .$$

Considering each group separately and expanding each term as a Taylor series about T_j^n, one gets

$$[0.5T_j^{n-1} - 2T_j^n + 1.5T_j^{n+1}]/\Delta t =$$
$$T_t + \Delta t\ T_{tt} + \frac{\Delta t^2}{6}T_{t^3} + \frac{\Delta t^3}{12}T_{t^4} + \frac{\Delta t^4}{120}T_{t^5} + \frac{\Delta t^5}{360}T_{t^6} + O(H)\ , \tag{1}$$

$$-\alpha(1+d)[T_{j-1} - 2T_j + T_{j+1}]^n/\Delta x^2 =$$
$$-\ \alpha(1+d)[T_{xx} + \frac{\Delta x^2}{12}T_{x^4} + \frac{\Delta x^4}{360}T_{x^6} + O(H)]\ , \tag{2}$$

$$\alpha d[T_{j-1} - 2T_j + T_{j+1}]^{n-1}/\Delta x^2 = \ \alpha d[T_{xx} + \frac{\Delta x^2}{12}T_{x^4}$$
$$+\frac{\Delta x^4}{360}T_{x^6} - \Delta t\ T_{x^2t} - \Delta t\frac{\Delta x^2}{12}T_{x^4t} + \frac{\Delta t^2}{2}T_{x^2t^2} + O(H)]\ . \tag{3}$$

Substituting (1), (2) and (3) in the given equation, one gets,

$$T_t - \alpha T_{xx} + \Delta t\ T_{tt} - \frac{\alpha\Delta x^2}{12}T_{x^4} - \alpha d\Delta t\ T_{xxt} + \frac{\Delta t^2}{6}T_{t^3}$$
$$-\ \alpha\frac{\Delta x^4}{360}T_{x^6} - \alpha d\ \Delta t\frac{\Delta x^2}{12}T_{x^4t} + \frac{\alpha d\Delta t^2}{2}T_{x^2t^2} + O(H) = 0\ . \tag{4}$$

Using the relations $\alpha\Delta t/\Delta x^2 = s$, and $T_{tt} = \alpha^2T_{x^4}$, $T_{xxt} = \alpha T_{x^4}$, $T_{t^3} = \alpha^3T_{x^6}$, etc (from 4.12),

the truncation error, E, is found to be,

$$E = s\alpha\Delta x^2 T_{x^4}\{1 - (1/12s) - d\}$$
$$+\ \alpha\Delta x^4\ T_{x^6}\,(s^2/6 - 1/360 - ds/12 + ds^2/2) + O(H)\ .$$

b) The scheme will be fourth-order accurate if

$$1-(1/12s) - d = 0 \quad \text{or} \quad d = 1-(1/12s)\ .$$

3.3 We consider the function, $y = \sin(\pi x/2)$. At $x = 0.5$ the exact solution for dy/dx is 1.11072. Discretising dy/dx as indicated by (a), (b) and (c) the following results are obtained.

Discretisation	dy/dx	Absolute error
(a) $(y_{j+1} - y_{j-1})/2\Delta x$	1.106159	0.456202×10^{-2}
(b) $(y_{j+1} - y_j)/\Delta x$	1.019102	0.916186×10^{-1}
(c) $(y_{j-2} - 8y_{j-1} + 8y_{j+1} - y_{j+2})/12\Delta x$	1.110699	0.224740×10^{-4}

Scheme (c) is seen to be the most accurate and (b) the least accurate. These results are broadly in agreement with Table 3.1.

3.4 Running the computer program developed for Problem 3.3 with different mesh sizes (i.e. Δx) gives the following results (in the tables $RRE \rightarrow$ Rate of Reduction in Error).

(a) **Discretisation:** $dy/dx \approx (y_{j+1} - y_{j-1})/2\Delta x$

Δx	Absolute error	RRE
0.1	0.45620×10^{-2}	—
0.05	0.11416×10^{-2}	≈ 4
0.025	0.28546×10^{-3}	≈ 4
0.0125	0.71368×10^{-4}	≈ 4

(b) **Discretisation:** $dy/dx \approx (y_{j+1} - y_j)/\Delta x$

Δx	Absolute error	RRE
0.1	0.91619×10^{-1}	—
0.05	0.44737×10^{-1}	2.05
0.025	0.22091×10^{-1}	2.03
0.0125	0.10975×10^{-1}	2.01

(c) **Discretisation:** $dy/dx \approx (y_{j-2} - 8y_{j-1} + 8y_{j+1} - y_{j+2})/\Delta x$

Δx	Absolute error	RRE
0.1	0.22474×10^{-4}	—
0.05	0.14077×10^{-5}	≈ 16
0.025	0.88033×10^{-7}	≈ 16
0.0125	0.55028×10^{-8}	≈ 16

Discussion:

1. As Δx is reduced, the error decreases for every scheme.

2. Again, scheme (c) turns out to be the most accurate.

3. The leading terms in the truncation error for the three schemes are (Table 3.3)

(a) $(\Delta x^2/6)T_{x^3}$, (b) $(\Delta x/2)T_{xx}$ and (c) $-(\Delta x^4/30)T_{x^5}$.

Hence, as the step size Δx is reduced by a factor of 2, we expect the absolute error to decrease by the following factors for the three schemes considered:

(a) 4, (b) 2 and (c) 16.

The computed errors follow this pattern closely as Δx is reduced.

3.5 We consider the function, $y = \sin(0.5\pi x)$. At $x = 0.5$, the exact value of d^2y/dx^2 is -1.74472. Discretising the term d^2y/dx^2 with schemes (a) and (b), the following results are obtained.

(a) **Discretisation:** $d^2y/dx^2 \approx (y_{j-1} - 2y_j + y_{j+1})/\Delta x^2$

Δx	d^2y/dx^2	Absolute error	RRE
0.1	-1.74113	0.35845×10^{-2}	—
0.05	-1.74382	0.89667×10^{-3}	≈ 4
0.025	-1.74450	0.22420×10^{-3}	≈ 4
0.0125	-1.74469	0.56053×10^{-4}	≈ 4

(b) **Discretisation:**

$$d^2y/dx^2 \approx (-y_{j-2} + 16y_{j-1} - 30y_j + 16y_{j+1} - y_{j+2})/12\Delta x^2.$$

Δx	d^2y/dx^2	Absolute error	RRE
0.1	-1.74470	0.11777×10^{-4}	—
0.05	-1.74472	0.73723×10^{-6}	≈ 16
0.025	-1.74472	0.46096×10^{-7}	≈ 16
0.0125	-1.74472	0.28813×10^{-8}	≈ 16

Discussion:

1. Scheme (b) is seen to be more accurate. Further, for each scheme, the error decreases as Δx is reduced.

2. The truncation error leading terms for the two schemes (Table 3.4) are

(a) $(\Delta x^2/12)T_{x^4}$ and (b) $-(\Delta x^4/90)T_{x^6}$.

3. As discussed for Problem 3.4, the errors for schemes (a) and (b) should decrease by a factor of 4 and 16 respectively when the mesh size is halved. The results presented above do exhibit this behaviour.

3.6 We start with the one-sided difference formula,

$$\partial T/\partial t = (T_j^{n+1} - T_j^n)/\Delta t \, . \tag{1}$$

But $T_j^n = \cos[m(x_j - qt^n)]$ and $T_j^{n+1} = \cos[m(x_j - qt^n) - mq\,\Delta t]$.

Therefore (1) can be written as

$$\partial T/\partial t = \{\cos[a - b] - \cos[a + b]\}/\Delta t \, , \tag{2}$$

where $a = m(x_j - qt^n) - mq\Delta t/2$ and $b = mq\ \Delta t/2$. (3)

This is equivalent to recognising that (1) is symmetric about $(x_j,\ t^{n+1/2})$.

Equation (2) can be simplified to

$$\begin{aligned}\partial T/\partial t &= (2/\Delta t)\sin(b)\sin(a)\\ &= (2/\Delta t)\sin(0.5mq\Delta t)\sin\Big\{m\left[x - q(t+\Delta t/2)\right]\Big\}\ ,\end{aligned} \tag{4}$$

as required.

For the three-term difference formula,

$$\begin{aligned}\partial T/\partial t &= (1.5T_j^{n+1} - 2T_j^n + 0.5T_j^{n-1})/\Delta t\\ &= 1.5(T_j^{n+1} - T_j^n)/\Delta t - 0.5(T_j^n - T_j^{n-1})/\Delta t\ .\end{aligned} \tag{5}$$

Using a similar strategy to that above, (5) can be written

$$\begin{aligned}\partial T/\partial t = \Big(&1.5\{\cos[a-b] - \cos[a+b]\}\\ &- 0.5\{\cos[c-b] - \cos[c+b]\}\Big)/\Delta t\ ,\end{aligned} \tag{6}$$

where a and b are given by (3) and $c = m(x_j - qt^n) + mq\Delta t/2$.

Equation (6) can be simplified to

$$\begin{aligned}\partial T/\partial t &= \Big[3\sin(a)\sin(b) - \sin(c)\sin(b)\Big]/\Delta t\\ &= (1/\Delta t)\sin(0.5mq\Delta t)\Big(3\sin\Big\{m\left[x - q(t+\Delta t/2)\right]\Big\}\\ &\quad - \sin\Big\{m\left[x - q(t-\Delta t/2)\right]\Big\}\Big)\ ,\end{aligned} \tag{7}$$

as required.

For Δt sufficiently small, sin(0.5mq)$\rightarrow$ 0.5mq Δt and cos(0.5mqΔt) $\rightarrow$ 1.0. Also we can write (4) as

$$\partial T/\partial t = (2/\Delta t)\sin(0.5mq\Delta t)\sin(d-b)\ , \tag{8}$$

where $d = \sin\{m(x-qt)\}$ and b is given by (3).

Applying the above limiting process, (8) becomes

$$\partial T/\partial t = mq\Big[\sin\{m(x-qt)\} - 0.5mq\Delta t\cos\{m(x-qt)\}\Big]\ . \tag{9}$$

In a similar manner, (7) becomes

$$\partial T/\partial t = mq\Big[\sin\{m(x-qt)\} - mq\Delta t\cos\ \{m(x-qt)\}\Big]\ . \tag{10}$$

As $\Delta t \rightarrow 0$, (9) and (10) coincide with the exact evaluation of $\partial \bar{T}/\partial x$.

3.7 For this problem, program DIFF is modified as follows

1. The following lines are inserted after line 19,

```
      S53=1.-5.*S/3.
      S1112=11.*S/12.
      S3=S/3.
      S12=S/12.
      S43=4.*S3
```

2. After line 25, insert `JMPM1 = JMAP-1`

3. Lines 57 and 58 are replaced by,

```
      DUM(2)=S53*TN(2) + S1112*TN(1) + 0.5*S*TN(3)
    1 + S3*TN(4) - S12*TN(5)
C
      DUM(JMAP)= S53*TN(JMAP) - S12*TN(JMAP-3)
    1  +S3*TN(JMAP-2) +0.5*S*TN(JMAP-1) + S1112*TN(JMAX)
C
      DO 7 J = 3,JMPM1
    7 DUM(J) = (1.-2.5*S)*TN(J) -S12*(TN(J-2)+TN(J+2))
    1 + S43*(TN(J-1)+TN(J+1))
```

Results:

Running the modified program with $\Delta x = 0.1$, the following rms solution errors are obtained. The corresponding errors for the FTCS scheme (i.e. running DIFF as it is) are also given.

s	rms error	rms error for FTCS scheme
0.3	0.8198	0.3618
0.2	0.5362	0.0868
0.1	0.2646	0.1814

For the present scheme the rms error is always higher than that for the FTCS scheme. The reasons for this behaviour will be discussed in the next chapter. (see Problem 4.1).

3.8 Program DIFF is to be modified so that, instead of the FTCS scheme, the following algorithm is used,

$$(0.5T_j^{n-1} - 2T_j^n + 1.5T_j^{n+1})/\Delta t - \alpha(T_{j-1}^n - 2T_j^n + T_{j+1}^n)/\Delta x^2 = 0\ .$$

The modifications to DIFF are effected as follows:

1. An extra array TPR of the same dimension as TN is introduced.

2. After line 19, insert,

```
      S215=2.*(1.-S)/1.5
      S15=S/1.5
      S515=0.5/1.5
```

3. After 49, insert,

```
      TPR(1) = 1
      TPR(JMAX) = 1
```

4. After 51, insert

```
      IF(T.LT.0.01)TPR(1) = 0.5
      IF(T.LT.0.01)TPR(JMAX) = 0.5
```

5. Lines 57-59 are replaced by,

```
      IF(T.LT.1.0E-05) THEN
      DO 7 J = 2,JMAP
      DUM(J) = (1.-2.*S)*TN(J) + S*(TN(J-1) + TN(J+1))
    7 CONTINUE
      ELSE
      DO 17 J=2,JMAP
   17 DUM(J)=S215*TN(J)+S15*(TN(J-1)+TN(J+1))-S515*TPR(J)
      ENDIF
```

6. After line 60, insert: `TPR(J) = TN(J)`

Results:

With $\Delta x = 0.1$, the rms solution errors computed for various values of s are as follows

s	rms error	rms error for FTCS scheme
0.5	2.462	0.9418
0.3	1.095	0.3618
0.1	0.1245	0.1814

The accuracy of the present scheme is inferior to that of the FTCS scheme. The next chapter (Problem 4.2) provides an explanation for this. As expected, the rms error reduces as s is reduced. The leading term in the truncation error (see Problem 3.2) is

$$s\alpha\Delta x^2 T_{x4}(1 - 1/12s - d) .$$

For the present scheme d = 0 and the truncation error is directly proportional to $s(s - 1/12)$. Hence, there is a reduction in rms error as s is reduced, as long as $s > 1/24$.

CTFD Solutions Manual: Chapter 4

Theoretical Background

4.1 Running the computer program developed for Problem 3.7, and DIFF the following solution errors are obtained when $s = 0.3$ and $t = 5000$ sec. In the table RRE denotes the Rate of Reduction of Error.

Δx	Scheme 3.7	FTCS Scheme
	rms error	rms error
0.2	1.634	0.59
0.1	0.4325 ($RRE \approx 4$)	0.187 (RRE ≈ 3)
0.05	0.1081 (RRE ≈ 4)	0.048 (RRE ≈ 4)

Discussion:

1. The leading term in the truncation error for the scheme considered in Problem 3.7 is proportional to Δx^2 (see Problem 4.3). Hence, as with the FTCS scheme, the solution error decreases by a factor of 4 as the mesh size is halved.

2. The accuracy of the present scheme is inferior to that of the FTCS scheme. This is despite the fact that the present scheme uses fourth-order accurate spatial differencing. The analysis carried out in Problem 4.3 shows that the time differencing employed for the present scheme produces a second-order error and this is the dominant contribution to the truncation error. This problem demonstrates that higher-order differencing for spatial derivatives alone is not sufficient to achieve a given accuracy.

3. It may be noted that the starting conditions used can also produce errors and influence the solution accuracy. The programs DIFF and the one written for the present problem use discrete starting conditions (typically 100 or 50 on the boundaries and 0 for the interior points). Such a discontinuous distribution of starting values will produce errors that are still contributing to the final error.

Starting the computations from $t = 75$ s, with the initial T values given by the exact solution for that time level, the following solution errors are obtained.

It may be noted that the errors are lower than when discontinuous starting conditions are employed.

$\Delta x =$	0.2	0.1	0.05
error	1.1113	0.2036	0.0655

4.2 Running the computer program developed for Problem 3.8, the following solution errors are obtained.

Δx	Scheme 3.8	FTCS Scheme
	rms error	rms error
0.2	1.762	0.59
0.1	0.3256 ($RRE \approx 4$)	0.187 (RRE ≈ 4)
0.05	0.079 (RRE ≈ 4)	0.048 (RRE ≈ 4)

Discussion:

1. The leading term in the truncation error for the present scheme is proportional to Δx^2 (see Problem 3.2) thus explaining the rate of reduction of error of 4.

2. The magnitude of the truncation error for the present scheme (see Problems 3.8 and 4.4) is somewhat higher than that for the FTCS scheme, contributing to the larger magnitudes of the solution errors.

4.3 Expanding every term of the scheme used in Problem 4.1 (see Problem 3.7 for details) as a Taylor series about T_j^n and rearranging the terms we have for interior points, and at j=2 and j=JMAX-1 respectively,

$$T_t - \alpha T_{xx} + \frac{\Delta t}{2}T_{tt} + \frac{\Delta t^2}{6}T_{t3} + \frac{1}{90}\alpha\Delta x^4\, T_{x6} + O(H) = 0\ , \tag{1}$$

$$T_t - \alpha T_{xx} + \frac{\Delta t}{2}T_{tt} + \frac{\Delta t^2}{6}T_{t3} + \frac{1}{12}\alpha\Delta x^3 T_{x5} + O(H) = 0\ , \tag{2}$$

$$T_t - \alpha T_{xx} + \frac{\Delta t}{2}T_{tt} + \frac{\Delta t^2}{6}T_{t3} - \frac{1}{12}\alpha\Delta x^3 T_{x5} + O(H) = 0\ . \tag{3}$$

For each of the cases 1, 2 and 3, the truncation error $E(\Delta x, \Delta t) \to 0$ as $\Delta x \to 0$ and $\Delta t \to 0$.

Hence the scheme is consistent with the governing equation,

$$\frac{\partial T}{\partial t} - \alpha\frac{\partial^2 T}{\partial x^2} = 0\ .$$

Order of truncation error:

Replacing the time derivatives in (1), (2) and (3) with the spatial derivatives (see 4.12) the leading term in the truncation error is found to be $0.5s\Delta x^2\alpha\ T_{x4}$ for each of the schemes, thus explaining the rate of decrease of the solution error observed in Problem 4.1.

Examining (1), (2) and (3) we find that the error caused by spatial discretisation is of third or fourth order. But it is the term $0.5\Delta t T_{tt}$ that is the leading term of the truncation error.

The improved accuracy of the FTCS scheme over that of the scheme considered in Problem 4.1 can be explained on the basis of the truncation error. These are $0.5\alpha\Delta x^2\,(s - 1/6)\,T_{x4}$ and $0.5s\Delta x^2\alpha T_{x4}$ respectively. The magnitude of the truncation error is lower for the FTCS scheme for every value of s. Further, whereas the FTCS scheme achieves a fourth order accuracy when $s = 1/6$, the present scheme cannot achieve better than second order accuracy for non-zero values of s.

4.4 It may be noted that the scheme considered in Problem 4.2 is the one studied in Problem 3.2 with $d = 0$. The leading term in the truncation error for this scheme is

$$s\alpha\ \Delta x^2\ T_{x4}\left(1 - \frac{1}{12s}\right)\ .$$

Again as $\Delta x \to 0$, $E \to 0$ (see Problem 3.2). Hence the scheme is consistent with the governing equation.

The leading term in the truncation error is proportional to Δx^2. Hence the observation (Problem 4.2) that the solution error decreases by a factor of 4 as the mesh size is halved is in agreement with the order of truncation error. However we would expect to achieve higher accuracy if $s = 1/12$.

4.5 Writing the 'error equations' for the scheme considered in Problem 4.1, the actual implementation of which is described in Problem 3.7, one gets for the interior points, and at j=2 and j=JMAX-1,

$$\xi_j^{n+1} = (1 - 2.5s)\xi_j^n - \frac{s}{12}\xi_{j-2}^n + \frac{4s}{3}\xi_{j-1}^n + \frac{4s}{3}\xi_{j+1}^n - \frac{s}{12}\xi_{j+2}^n\ , \tag{1}$$

$$\xi_j^{n+1} = \left(1 - \frac{5s}{3}\right)\xi_j^n + \frac{11s}{12}\xi_{j-1}^n + \frac{s}{2}\xi_{j+1}^n + \frac{s}{3}\xi_{j+2}^n - \frac{s}{12}\xi_{j+3}^n\ , \tag{2}$$

$$\xi_j^{n+1} = \left(1 - \frac{5s}{3}\right)\xi_j^n + \frac{11s}{12}\xi_{j+1}^n + \frac{s}{2}\xi_{j-1}^n + \frac{s}{3}\xi_{j-2}^n - \frac{s}{12}\xi_{j-3}^n\ . \tag{3}$$

Let $\xi_j^n = G^n e^{i\theta j}$. Substituting for ξ_j^n, ξ_j^{n+1} etc. in the above equations and simplifying one gets for the interior points, and the boundary points j=2 and j=JMAX-1,

$$G = 1 - 2.5s + \frac{s}{6} - \frac{s}{3}\cos^2\theta + \frac{8s}{3}\cos\theta, \tag{4}$$

$$\begin{aligned} G = 1 &- \frac{5}{3}s + \frac{17s}{12}\cos\theta + \frac{s}{3}\cos 2\theta - \frac{s}{12}\cos 3\theta \\ &+ i\left\{-\frac{5s}{12}\sin\theta + \frac{s}{3}\sin 2\theta - \frac{s}{12}\sin 3\theta\right\} \end{aligned} \tag{5}$$

and

$$\begin{aligned} G = 1 &- \frac{5}{3}s + \frac{17s}{12}\cos\theta + \frac{s}{3}\cos 2\theta - \frac{s}{12}\cos 3\theta \\ &+ i\left\{\frac{5s}{12}\sin\theta - \frac{s}{3}\sin 2\theta + \frac{s}{12}\sin 3\theta\right\} . \end{aligned} \tag{6}$$

On running a computer program to calculate the absolute value of G for various values of s and θ, we find that the interior point scheme, (4), is stable for all $s \leq 3/8$. The boundary schemes, (5) and (6), are stable for all $s \leq 0.75$. Hence for the overall scheme the stability restriction is $s \leq 3/8$.

Note: The stability conditions can be derived from (5) or (6) directly using complex algebra. It is not always necessary to use a computer.

4.6 The 'error equation' for the scheme in Problem 4.2 is

$$0.5\xi_j^{n-1} - 2\,\xi_j^n + 1.5\,\xi_j^{n+1} = s(\xi_{j-1}^n - 2\xi_j^n + \xi_{j+1}^n) \ . \tag{1}$$

With the substitution that $\xi_j^n = Ge^{i\theta j}$, one gets

$$\frac{1}{2G} - 2 + 1.5G = s(e^{-i\theta} - 2 + e^{i\theta})$$

$$\text{or} \quad 3G^2 + G(-4 - 4s\cos\theta + 4s) + 1 = 0 \ . \tag{2}$$

Solving (2) for G, we find that $s \leq 1.0$ for stability.

4.7 The stability restrictions on the schemes considered in Problems 4.1 and 4.2 are found to be $s \leq 3/8$ and $s \leq 1.0$ respectively. Running the program with values of s close to these limits the following results are obtained for $\Delta x = 0.05$, $t = 75000$ sec.

Scheme in Problem 4.1

$s =$	0.374	0.38	0.4
error	0.0015	0.2573	0.83×10^{33}

Scheme in Problem 4.2

$s =$	0.99	1.0	1.1
error	0.0075	0.0078	$.. \times 10^{34}$

These findings confirm the stability limits established by the von Neumann method.

4.8 The scheme obtained by a forward time representation for $\partial T/\partial t$ and $\partial^2 T/\partial x^2 = aT_j + bT_{j+1} + cT_{j+2} + dT_{j+3}$ is

$$T_j^{n+1} = T_j^n(1 + as) + bs\,T_{j+1}^n + cs\,T_{j+2}^n + ds\,T_{j+3}^n \ . \tag{1}$$

Following the procedure outlined before, one gets

$$\begin{aligned} G =&(1 + as) + bs(\cos\theta + i\sin\theta) + cs(\cos 2\theta + i\sin 2\theta) \\ &+ ds(\cos 3\theta + i\ \sin 3\theta) \ . \end{aligned} \tag{2}$$

Substituting the values for a, b, c and d, we have

$$\begin{aligned} G = &(1 + 2s) + s(-5\cos\theta + 4\ \cos 2\theta - \cos 3\theta) \\ &+ is(-5\sin\theta + 4\ \sin 2\theta - \sin 3\theta) \ . \end{aligned} \tag{3}$$

Working out the absolute value of G for various s and θ values shows that the scheme is unstable for all $s > 0$.

4.9 The given scheme is

$$M_x \frac{\Delta T_j^{n+1}}{\Delta t} - \alpha\left[\beta\, L_{xx}\, T_j^{n+1} + (1 - \beta)L_{xx}\, T_j^n\right] = 0 \ , \tag{1}$$

where $M_x = (\delta$, 1-2δ, $\delta)$ and $L_{xx}\ T_j = (T_{j-1} - 2T_j + T_{j+1})/\Delta x^2$,

Substituting for M_x and L_{xx} in (1), one gets

$$\begin{aligned} &\frac{1}{\Delta t}\Big\{\left[\delta T_{j-1} + (1 - 2\delta)T_j + \delta T_{j+1}\right]^{n+1} \\ &- \left[\delta T_{j-1} + (1 - 2\delta)T_j + \delta T_{j+1}\right]^n\Big\} \\ &- \frac{\alpha}{\Delta x^2}\Big\{\beta\left[T_{j+1} - 2T_j + T_{j-1}\right]^{n+1} \\ &+ (1 - \beta)\left[T_{j+1} - 2T_j + T_{j-1}\right]^n\Big\} = 0 \ . \end{aligned} \tag{2}$$

Substituting for $\alpha\Delta t/\Delta x^2$ (i.e. s) and simplifying, we have

$$\begin{aligned} &T_{j-1}^{n+1}(\delta - s\beta) + T_j^{n+1}(1 - 2\delta + 2\ s\beta) + T_{j+1}^{n+1}(\delta - s\beta) \\ &= T_{j-1}^n\{\delta + s(1 - \beta)\} + T_j^n\{1 - 2\delta - 2s(1 - \beta)\} + \\ &T_{j+1}^n\{\delta + s(1 - \beta)\}. \end{aligned} \tag{3}$$

Let $\delta_1 = \delta - s\beta$, $\delta_2 = \delta + s(1 - \beta)$.

Then (3) simplifies to

$$\begin{aligned}&\delta_1\ T_{j-1}^{n+1} + (1-2\ \delta_1)T_j^{n+1} + \delta_1\ T_{j+1}^{n+1} = \\ &\delta_2\ T_{j-1}^{n} + (1-2\ \delta_2)T_j^{n} + \delta_2\ T_{j+1}^{n}\ .\end{aligned} \tag{4}$$

This is a tridiagonal system whose stability can be readily determined by the matrix method. Equation (4) is of the form $\underline{\mathbf{A}}\xi^{n+1} = \underline{\mathbf{B}}\xi^n$ (see 4.25), where

$$\underline{\mathbf{A}} = \begin{bmatrix} 1-2\delta_1 & \delta_1 & & & \\ \delta_1 & 1-2\delta_1 & \delta_1 & & \\ & \delta_1 & 1-2\delta_1 & \delta_1 & \\ - & - & - & - & - \\ & & & \delta_1 & 1-2\delta_1 \end{bmatrix}$$

and

$$\underline{\mathbf{B}} = \begin{bmatrix} 1-2\delta_2 & \delta_2 & & & \\ \delta_2 & 1-2\delta_2 & \delta_2 & & \\ & \delta_2 & 1-2\delta_2 & \delta_2 & \\ - & - & - & - & - \\ & & & \delta_2 & 1-2\delta_2 \end{bmatrix} .$$

The scheme (4) is stable if

$$|\lambda_B{}^m/\lambda_A{}^m| \le 1\ .$$

From (4.26) we have

$\lambda_A = 1 - 2\delta_1 - 2\delta_1 \cos(j\pi/(j-1))$ and $(\lambda_A)_{\max} = 1 - 4\delta_1$.
Similarly, $(\lambda_B)_{\max} = 1 - 4\delta_2$.

For stability,

$$\left|\frac{1-4\delta_2}{1-4\delta_1}\right| \le 1\ . \tag{5}$$

This condition is met for all $s \le (0.5 - 2\delta)/(1-2\beta)$.

Remarks:

It remains to check whether the scheme (1) is consistent with the governing equation. Expanding every term of (1) as a Taylor series about $T_j{}^n$ we have

$$\begin{aligned}T_t - \alpha T_{xx} = &-\delta_1\left[\frac{1}{12}\frac{\alpha}{s}\Delta x^2\ T_{x^4} + \Delta x^2 T_{x^2 t} + \frac{1}{12}\Delta x^4\ T_{x^4 t}\right.\\ &\left.+\frac{\Delta t\Delta x^4}{24}T_{x^4 t^2} + \frac{\Delta t}{3}T_{t^3}\right] - \frac{\Delta t}{2}T_{tt} - \frac{\Delta t^2}{6}T_{t^3} - \frac{\Delta t^3}{24}T_{t^4} - 2\delta_1\frac{\Delta t^2}{6}T_{t^3}\\ &+\delta_2\ \Delta x^2\ T_{xx} + \frac{1}{12}\delta_2\frac{\alpha}{s}\Delta x^3\ T_{x^4} + O(H)\ .\end{aligned}$$

We see that the $RHS \to 0$ as $\Delta x \to 0$ and $\Delta t \to 0$.

Hence the scheme is consistent with the governing equation.

4.10 The scheme considered in Problem 4.1 has a truncation error given by (see Problem 4.3).

$$\frac{s\Delta x^2}{2}\alpha\ T_{x^4} + O(H). \tag{1}$$

Let us carry out a Richardson extrapolation on two grids, $\Delta x_a = 0.1$ and $\Delta x_b = 0.05$.

From (4.45) we have $a = -1/3$, $b = 4/3$.

The Richardson extrapolation is carried out in the following manner -

1. The solution S_a on a coarse grid Δx_a is first computed to the desired time level.
2. The solution S_b on a fine grid Δx_b is similarly computed.
3. The extrapolated solution is given by

$S = aS_a + bS_b$ .... (see 4.44).

The rms solution errors computed are given in the following table. The rate of reduction of the error is indicated in brackets.

s = 0.3

Scheme\ $\Delta x =$	0.2	0.1	0.05	0.025
Problem 4.1	1.634	0.4325(≈ 4)	0.1081(≈ 4)	0.02731(≈ 4)
Problem 4.1 + *RE*	1.394	0.1226(≈ 11)	0.03293(≈ 4)	

s = 0.2

Scheme\ $\Delta x =$	0.2	0.1	0.05	0.025
Problem 4.1	1.282	0.3014(≈ 4)	0.0727	0.0182(≈ 4)
Problem 4.1 + *RE*	0.903	0.0083(≈ 100)	0.0003(≈ 30)	

s = 0.1

Scheme\ $\Delta x =$	0.2	0.1	0.05	0.025
Problem 4.1	0.8494	0.1579(≈ 5)	0.03654(≈ 4)	0.01666(≈ 2)
Problem 4.1 + *RE*	0.2583	0.00734(≈ 35)	0.00022(≈ 34)	

Remarks:

The scheme without Richardson extrapolation gives a convergence rate of about 4 as expected. With extrapolation the convergence rate increases to above 30 (except when $s = 0.3$) indicating that now the error is proportional to Δx^4 or some higher power of Δx. Since Richardson extrapolation relies on cancellation of terms in the truncation error it is not surprising that a relatively fine grid is required to obtain consistent results from Richardson extrapolation.
The code developed for Problem 3.7 is modified as follows to implement Richardson extrapolation. Two additional arrays $TCORSE$ and $TFINE$, each having the same dimension as TN, are introduced. The following lines are inserted immediately after the READ statement.

```
      DO 111 NRICH=1,2
      IF(NRICH.EQ.1) JCORSE=JMAX
      IF(NRICH.EQ.2) JMAX=2*JMAX-1
```

After the instruction, `T = T + DELT`, the following lines are introduced.

```
      N = N + 1
      WRITE(6,10)T,(TD(J),J=1,JMAX)
   10 FORMAT(' T= ',F5.0,'  TD=',11F6.2)
C
      IF(N .GE. NMAX)GOTO 133
      IF(T .LT. TMAX)GOTO 6
C
  133 CONTINUE
      IF(NRICH.EQ.1) THEN
      WRITE(6,*)'NO OF COARSE TIME STEPS>',N
      DO 123 J=1,JMAX
  123 CORSE(J)=TD(J)
      ELSE
      DO 124 J=1,JMAX
  124 TFINE(J)=TD(J)
      ENDIF
  111 CONTINUE
      CALL RICHRD(TCORSE,TFINE,TD,JCORSE,JMAX)
      JMAX=JCORSE
```

RICHRD is the subroutine developed to carry out the Richardson extrapolation and its listing is as follows.

```
      Subroutine RICHRD(TCORSE,TFINE,TD,JCORSE,JMAX)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION TCORSE(41),TFINE(41),TD(41),TEMP(41)
C
```

```
      C1=-1./3.
      C2=4./3.
C
      JJ=0
      DO 10 J=1,JMAX,2
      JJ=JJ+1
      TEMP(JJ)=TFINE(J)
   10 CONTINUE
      DO 12 J=1,JCORSE
      TD(J)=C1*TCORSE(J)+C2*TEMP(J)
   12 CONTINUE
      RETURN
      END
```

4.11 Carrying out Richardson extrapolation for scheme in Problem 4.2, the following rms solution errors are obtained.

s = 0.3

Scheme \ $\Delta x =$	0.2	0.1	0.05	0.025
Problem 4.2	1.762	0.3256(≈ 5)	0.07896(≈ 4)	0.01979(≈ 4)
Problem 4.2 + *RE*	1.34	0.1185(≈ 11)	0.03257(≈ 4)	

s = 0.2

Scheme\ $\Delta x =$	0.2	0.1	0.05	0.025
Problem 4.2	0.8102	0.1599(≈ 5)	0.03753(≈ 4)	0.00934(≈ 4)
Problem 4.2 + *RE*	0.9225	0.0060(≈ 153)	0.00027(≈ 22)	

s = 0.1

Scheme \ $\Delta x =$	0.2	0.1	0.05	0.025
Problem 4.2	0.3018	0.0865(≈ 3.5)	0.02258(≈ 4)	0.00769(≈ 3)
Problem 4.2 + *RE*	0.3141	0.0010(≈ 312)	0.00022(≈ 4)	

For a sufficiently fine grid Richardson extrapolation produces a more accurate solution. The improvement in accuracy is greater for smaller values of s.
The modifications to the program developed for Problem 3.8 to implement Richardson interpolation closely follow those carried out for Problem 4.10. Again subroutine RICHRD is employed.

4.12 The scheme used in Problem 4.10 employs the Richardson extrapolation with the scheme used in Problem 3.7 and 4.1. Discussion of this problem follows the changes to program DIFF for Problem 3.7 listed in Chapter 3.

The operation count is

$$\{23\ FL + 17\ FX + 9R + 2L + (N_x - 4)(11\ FX + 10\ FL + 2R)\}N_t\ ,$$

where N_x is the number of grid points used and N_t is the number of time steps taken. Assuming (see Section 4.5.1),

$$1FL_{eq} = 20\ FX = 10R = 10L = 0.1M\ ,$$

we have an operation count equal to

$$\{25 + (N_x - 4)(10.75)\}N_t \cdot FL_{eq}\ . \tag{1}$$

With Richardson extrapolation the operation count is

$$\begin{aligned}&\Big[\{25 + 10.75(N_x - 4)\}N_t\Big]_{coarse} FL_{eq}\\ &+ \Big[\{25 + 10.75(N_x - 4)\}N_t\Big]_{fine} FL_{eq}\ .\end{aligned} \tag{2}$$

A comparison of the FTCS and the present scheme can be made as follows. Using a Δx of 0.05, it is found that the FTCS and the present scheme achieve accuracies of 0.7173×10^{-1} and 0.7271×10^{-1}, the values of s being 0.37 and 0.2 respectively. The number of time steps taken are 55 and 100. The operation counts are

FTCS Scheme: $[4 + (N_x - 2)5.8]N_t = 6,281\ FL_{eq}$ { see p. 94 of the text }.

Present Scheme: $20,775\ FL_{eq}$.

With Richardson extrapolation and $\Delta x = 0.05$ the FTCS and the present scheme achieve accuracies of 0.248×10^{-3} and 0.266×10^{-3} when the s values are 0.4 and 0.2 respectively. The number of time steps taken are 50 and 200 on the coarse and the fine grids for the FTCS and 100 and 400 for the present scheme. The operation counts are:

FTCS Scheme:	51,750 FL_{eq}.
Present Scheme :	200,000 FL_{eq}.

Thus, we find that the present scheme is only about half as efficient as the FTCS scheme, when Richardson extrapolation is applied to both schemes. However compared with the FTCS scheme without Richardson extrapolation, the addition of Richardson extrapolation to either scheme brings about a substantial increase in computational efficiency.

4.13 The operation counts for the scheme, also used in Problems 3.8, 4.2, 4.11, are,

$$\begin{aligned}\Big\{&N_t(10\ FX + 3FL + 9R + 4L)\\ &+ N_t(1FX + 11)\\ &+ (7FX + 5FL + 1R)\\ &+ (N_t - 1)(5FX + 6FL + R)\\ &+ N_t(2FX + FL + R)\\ &+ N_t(N_x - 2)(3FL + FX + R)\\ &+ (N_t - 1)(N_x - 2)(4FX + 6FL + R)\Big\}FL_{eq}\ ,\end{aligned}$$

which simplifies to $\{11.55N_t + (N_x - 2)(9.3\ N_t - 6.15)\}FL_{eq}$.

(a) The present scheme with $\Delta x = 0.05$ and $s = 0.3$ achieves an accuracy of 0.789×10^{-1} compared with FTCS which achieves an accuracy of 0.7173×10^{-1} with $s = 0.05$ and $s = 0.3$. The number of time steps taken are 67 and 55 respectively. The operation counts are:

FTCS Scheme:	6,281 FL_{eq}.
Present scheme:	$\approx 12,500 FL_{eq}$.

(b) With Richardson extrapolation the present scheme achieves an accuracy of 0.2738×10^{-3} as against the FTCS scheme which achieves an accuracy of 0.248×10^{-3} (see Problem 4.12 for other details). For this example $\Delta x = 0.05$ and $s = 0.2$. The number of time steps taken are 100 and 400 on the coarse and the fine grids respectively. The operation counts are:

FTCS Scheme:	51,750 FL_{eq}.
Present scheme:	150,000 FL_{eq}.

Thus this scheme is not as efficient as the FTCS scheme with Richardson extrapolation but it is more efficient than the scheme considered in Problem 4.10.

4.14 To judge which is the best scheme, computational efficiency, stability restrictions and the ease or otherwise of coding have to be considered. Stability restrictions have already been deduced in Problems 4.5 and 4.6 for the schemes introduced in Problems 4.1 and 4.2. For the FTCS scheme, the stability condition is $s \leq 0.5$ (see 4.37).

It remains to calculate the computational efficiencies. From Sections 4.5 and 4.4.1, we find that the computational efficiency can be expressed as

$$CPE = s\ \Delta x^3/\alpha\epsilon\ .$$

For comparison, the behaviour of the schemes at $s = 0.3$, $\Delta x = 0.05$, $\alpha = 0.1 \times 10^{-4}$ and $t = 5000$ sec. is selected. Noting that the solution errors are 0.04797, 0.1081 and 0.07896 (see Problems 4.1 and 4.2), the computational efficiencies are given by $a/.04797$, $a/0.1081$, $a/0.07896$, where $a = 0.3 \times (0.05)^3/0.1 \times 10^{-4}$ and are 78.1, 34.6 and 47.49.

Summarising the findings in a table, we have

	FTCS	Scheme in Problem 4.1	Scheme in Problem 4.2
Comp. efficiency	78.1	34.6	47.49
Stability restriction	$s \le \frac{1}{2}$	$s \le \frac{3}{8}$	$s \le 1.0$
Ease of coding	easiest	more difficult than FTCS	most difficult of the three

Evidently, based on the above criteria, the FTCS scheme is to be judged the 'best' scheme.

CTFD Solutions Manual: Chapter 5

Weighted Residual Methods

5.1 Applying the WRM methods described in Sect. 5.1 produces the following coefficients in the approximate solution,

Coefficient	a_1	a_2
Galerkin	2.912378	-1.690000
Subdomain	2.577660	-1.408464
Least-squares	2.558151	-1.412076

The corresponding solutions are

x	Galerkin	Subdomain	Least-squares	Exact
0.2	0.50889	0.45384	0.44998	0.50000
0.4	0.84770	0.76288	0.75598	0.86603
0.6	0.98207	0.89500	0.88699	1.00000
0.8	0.87763	0.81808	0.81200	0.86603
1.0	0.50000	0.50000	0.50000	0.50000
error	0.01205	0.06595	0.07107	

Clearly for this problem the Galerkin formulation is most accurate. However all three methods are reasonably accurate since only two coefficients are to be chosen.

5.2 Application of the method of weighted residuals (Sect. 5.1) to the diffusion equation produces a system of ordinary differential equation,

$$\underline{M}\dot{A} + \underline{B}A + C = 0, \quad (1)$$

where an element of $\dot{\mathbf{A}}$ is da_j/dt and an element of $\underline{\mathbf{M}}$ is

$$m_{kj} = \int_0^1 W_k\ \phi_j\ dx$$

and $\phi_j = x^j - x^{j+1}$.

The elements of $\underline{\mathbf{B}}$ and C are

$$b_{kj} = -\int_0^1 W_k \frac{d^2\phi_j}{dx^2}dx, \quad c_k = -\int_0^1 W_k \frac{d^2\theta}{dx^2}(x,0)dx.$$

The form of W_k depends on the particular method (Sect. 5.1). For the Galerkin method,

$$m_{kj} = 1/(j+k+1) - 2/(j+k+2) + 1/(j+k+3)$$

and $b_{kj} = -\left[j(j-1)/(j+k-1) - 2j^2/(j+k) + j(j+1)/(j+k+1)\right]$

Equation (1) is integrated in time using a Euler scheme with $dt = 0.001$. Typical solutions for $N = 3$ are shown in Table 1. The solution θ_b is equivalent to $\bar{\theta}$ except that the time dependent part of solution is advanced by the Euler scheme. Thus the rms errors in Table 2 are based on $\theta - \theta_b$ since this isolates the error caused by the weighted residual method (also see Sect. 5.6.1). It is apparent that the accuracy of all methods increases rapidly with increasing N. The Galerkin method is most accurate and has the highest convergence rate.

Table 1 **Numerical solutions at $x = 0.5$ for $N = 3$**

t	Galerkin method	Subdomain method	Collocation method	Exact $\bar{\theta}$	Approx θ_b
0	1.5000	1.5000	1.5000	1.5000	1.5000
0.04	1.1728	1.1723	1.1716	1.1738	1.1725
0.08	0.9527	0.9513	0.9529	0.9540	0.9523
0.12	0.8084	0.8023	0.8058	0.8059	0.8041
0.16	0.7053	0.7018	0.7072	0.7062	0.7045
0.20	0.6383	0.6341	0.6412	0.6389	0.6376

Table 2 $||\theta - \theta_b||_{rms}$ **variation with N at $t = 0.20$**

N	Galerkin method	Subdomain method	Collocation method
3	0.471×10^{-3}	0.319×10^{-2}	0.431×10^{-2}
5	0.521×10^{-5}	0.484×10^{-4}	0.999×10^{-4}
7	0.396×10^{-7}	0.536×10^{-6}	0.147×10^{-5}

5.3 Substitution of the approximate solution into the governing equation produces the following residual ($N = 3$),

$$R = F_1a_1 + F_2a_2 + F_3a_3 + 1.0$$

where
$$F_1 = -2\{(1-x^2) + (b/a)^2(1-y^2)\}$$
$$F_2 = -4\{(1-x^2)^2(1-3y^2) + (b/a)^2(1-3x^2)(1-y^2)^2\}$$
$$F_3 = -6\{(1-x^2)^3(1-y^2)(1-5y^2) + (b/a)^2(1-x^2)(1-5x^2)(1-y^2)^3\}.$$

Algebraic equations are obtained from

$$\int_{-1}^1\int_{-1}^1 RW_k dx dy = 0,$$

where the form of W_k depends on the particular method. The system of equations can be written

$$\underline{\mathbf{M}}\mathbf{A} = \mathbf{C},$$

where the coefficients of $\underline{\mathbf{M}}$ and $\mathbf{C}$ are determined by the method of weighted residuals and $\mathbf{A}$ is the vector of unknown coefficients. Substitution into the approximate solution leads to the rms errors shown in the table,

Method	N = 1	N = 2	N = 3
Galerkin	0.00838	0.00231	0.00197
Subdomain	0.04490	0.00812	0.00377
Collocation	0.01828	0.00633	0.00386
Linear		6x6 grid	11x11 grid
FEM		0.00724	0.00157

The traditional Galerkin method is more accurate than the subdomain or collocation methods. The linear finite element method on an 11 × 11 grid has 81 nodal unknowns. The accuracy is not much greater than the accuracy of the traditional Galerkin method with three degrees of freedom.

5.4 The rms errors ($\Delta\phi_{rms} = ||\phi - \bar{\phi}||_{rms}$) and number of iterations to convergence (N_{it}) are compared for increasing r_x in the following table. Increasing r_x causes the region and the individual volumes to become distorted. This also causes a change in the character of the exact solution since the grid is constructed by taking equal intervals of θ. Thus for $r_x = 8.0$ the solution changes very rapidly close to corner Y in Fig. 5.4.

	$r_x = 1.0$		$r_x = 4.0$		$r_x = 8.0$	
grid	$\Delta\phi_{rms}$	N_{it}	$\Delta\phi_{rms}$	N_{it}	$\Delta\phi_{rms}$	N_{it}
6x6	0.1326	15	0.0725	15	0.1287	16
11x11	0.0471	19	0.0600	19	0.0869	16
21x21	0.0138	51	0.0340	61	0.0558	61

Generally grid refinement improves the accuracy but not to the same extent as for $r_x = 1.0$. Also increasing r_x usually leads to a reduction in accuracy due to increased grid distortion. The result for $r_x = 4.0$, JMAX $= KMAX = 6$ is due to compensating changes in the exact solution as noted above. The iterative rate of convergence is relatively insensitive to increasing the value of r_x.

5.5 The discretisation of $\partial\bar{\phi}/\partial x + \partial\bar{\phi}/\partial y - S$ leads to the addition of the following terms to the left-hand side of (5.40)

$$(1/\alpha)[\mathcal{A}_{j,k}\ S_{j,k} - 0.5\{(\Delta y_{DA} - \Delta x_{DA})\phi_{j-1,k} + (\Delta y_{BC} - \Delta x_{BC})\phi_{j+1,k}$$

$$+(\Delta y_{AB} - \Delta x_{AB})\phi_{j,k-1} + (\Delta y_{CD} - \Delta x_{CD})\phi_{j,k+1}\}], \qquad (1)$$

where

$$\mathcal{A}_{j,k} = 0.5[(x_A - x_D)(y_B - y_A) + (x_B - x_C)(y_C - y_D)$$
$$+(x_B - x_A)(y_D - y_A) - (x_C - x_D)(y_C - y_B)].$$

$S_{j,k}$ is evaluated from the following code after line 51

$C2K = CK * CK - SK * SK$
$S2K = 2. * SK * CK$
$SC(J,K) = (C2K - S2K)/R/R$

After line 121, $\mathcal{A}_{j,k}$ and $(\Delta y_{DA} - \Delta x_{DA})$ etc. are evaluated from

$AR(J,K) = 0.5(DXD * DYA + DXB * DYC - DXA * DYD - DXC * DYB)$
DYXAB$(J,K) = DYA - DXA$
DYXBC$(J,K) = DYB - DXB$
DYXCD$(J,K) = DYC - DXC$
DYXDA$(J,K) = DYD - DXD$

After line 142, (1) is added to the residual as

DRES $= DYXDA(J,K) * PHI(JM,K) + DYXBC(J,K) * PHI(JP,K)$
$+DYXAB(J,K) * PHI(J,KM) + DYXCD(J,K) * PHI(J,KP)$
$PHD = PHD + (AR(J,K) * SC(J,K) - 0.5*$DRES$)/ALPH$

The solution behaviour is indicated in the following table

	$\alpha = 1.0, \lambda = 1.5$		$\alpha = 0.01, \lambda = 0.2$	
grid	$\Delta\phi_{rms}$	N_{it}	$\Delta\phi_{rms}$	N_{it}
6x6	0.1627	15	0.7794	62
11x11	0.0556	20	0.2560	96
21x21	0.0161	56	0.0752	202

The equation being solved is a convection/diffusion equation (Sect. 9.3) in two dimensions. For $\alpha = 1$ the convergence behaviour and residual levels are similar to that shown in Table 5.5. However for $\alpha = 0.01$, the convective terms are dominant and it is necessary to use a relaxation parameter, $\lambda = 0.2$. As a result many more iterations are required to reach convergence. Although the solution becomes more accurate as the grid is refined, the accuracy is less than for the $\alpha = 1.0$ case. This behaviour is typical of convection dominated equations.

5.6 Evaluation of (5.33) produces an equation, equivalent to (5.34), but based on the volume $D'C'CD$ with point $(j,1)$ at the midpoint of $D'C'$.

The following discretisations are assumed,

$$\partial\phi/\partial x|_{j+1/2,k} = (\phi_{j+1,k} - \phi_{j,k})/(x_{j+1,k} - x_{j,k})$$
$$\partial\phi/\partial x|_{j-1/2,k} = (\phi_{j,k} - \phi_{j-1,k})/(x_{j,k} - x_{j-1,k}).$$

The evaluation of $\partial\phi/\partial y$ on DD', $D'C'$ and $C'C$ are determined by the given boundary condition evaluated at $k = 1$.

Consequently the following equation replaces (5.39)

$$-\frac{\Delta x_{D'C'}}{r^2_{j,1}} + \frac{\Delta y_{C'C}}{\Delta x_{j+1/2}}(\phi_{j+1,1} - \phi_{j,1}) - \frac{\Delta x_{C'C}}{r^2_{j+1/2,1}} + Q_{CD}(\phi_{j,2} - \phi_{j,1})$$

$$+P_{CD}(\phi_D - \phi_C) + \frac{\Delta y_{DD'}}{\Delta x_{j-1/2}}(\phi_{j,1} - \phi_{j-1,1}) - \frac{\Delta x_{DD'}}{r^2_{j-1/2,1}} = 0,$$

where $\Delta x_{j-1/2} = x_{j,1} - x_{j-1,1}$ and $\Delta x_{j+1/2} = x_{j+1,1} - x_{j,1}$.

By substituting for ϕ_C and ϕ_D, the following algorithm is obtained in place of (5.41) on $WX(k = 1)$,

$$\phi^*_{j,1} = \Bigg[-0.25P_{CD}\phi_{j-1,2} + Q_{CD}\phi_{j,2} + 0.25P_{CD}\phi_{j+1,2}$$

$$-\left\{0.25P_{CD} + \frac{\Delta y_{DD'}}{\Delta x_{j-1/2}}\right\}\phi_{j-1,1} + \left\{0.25P_{CD} + \frac{\Delta y_{CC'}}{\Delta x_{j+1/2}}\right\}\phi_{j+1,1}$$

$$-\frac{\Delta x_{D'C'}}{r^2_{j,1}} - \frac{\Delta x_{C'C}}{r^2_{j+1/2,1}} - \frac{\Delta x_{DD'}}{r^2_{j-1/2,1}}\Bigg] \Big/ \left[Q_{CD} + \frac{\Delta y_{C'C}}{\Delta x_{j+1/2}} - \frac{\Delta y_{DD'}}{\Delta x_j}\right].$$

The evaluation of this equation can be carried out after line 148.

Grid	$\|\|\phi - \bar{\phi}\|\|_{rms}$	No. of iterations to convergence
6x6	0.1525	15
11x11	0.0611	41
21x21	0.0183	102

As the above table indicates, the accuracies are slightly less than for corresponding Dirichlet boundary conditions (Table 5.5), and the errors are largest at $k = 1$. Also the number of iterations to reach convergence increases more rapidly with grid refinement than for Dirichlet boundary conditions.

5.7 The rms errors for linear and quadratic finite element interpolation are indicated in the following table.

Linear interpolation		Quadratic interpolation	
Number of elements, N	rms error	Number of elements, N	rms error
2	0.0911	1	0.0819
4	0.0242	2	0.00974
8	0.00611	4	0.00115
16	0.00153	8	0.000144

These results may be compared with Table 5.6. It is clear that linear interpolation is converging like N^{-2} and quadratic interpolation is converging like N^{-3}. This behaviour is in agreement with theoretical results.

5.8 Linear and quadratic interpolation is applied to

$$\bar{y} = (1 + \sin \pi x) \exp(-2x),$$

on a geometrically growing grid ($r_x = 1.20$). The results are summarised below. Each pair of cases has the same domain ($0 \le x \le x_{max}$) and the same number of nodes (N_x). For most cases the use of a non-uniform grid ($r_x = 1.2$) allows more points where the solution is changing most rapidly (small x). This leads to a smaller maximum error (E_{max}) and rms error (E_{rms}). However the advantage, for the present function, diminishes as the grid is refined.

Linear interpolation

Δx	r_x	x_{max}	N_x	E_{rms}	E_{max}	x_{Emax}
0.20	1.20	1.29	5	0.0218	0.055	0.12
0.32	1.00	1.29	5	0.0360	0.091	0.16
0.10	1.20	1.19	7	0.0120	0.024	1.04
0.20	1.00	1.19	7	0.0143	0.039	0.10
0.05	1.20	1.25	10	0.0073	0.0158	1.12
0.125	1.00	1.25	10	0.0069	0.0199	0.07

Quadratic interpolation

Δx	r_x	x_{max}	N_x	E_{rms}	E_{max}	x_{Emax}
0.20	1.20	1.29	5	0.0105	0.0226	0.42
0.32	1.00	1.29	5	0.0201	0.0395	0.50
0.10	1.20	1.19	7	0.0032	0.0080	0.35
0.20	1.00	1.19	7	0.0043	0.0092	0.32
0.05	1.20	1.56	11	0.0026	0.0055	1.11
0.16	1.00	1.56	11	0.0019	0.0043	0.37

5.9 Bilinear and biquadratic finite element interpolation are applied to

$\bar{y} = \{1 + \sin(\pi x)\} \exp(-2x) \sin(\pi y)$.

The effect of grid refinement on the rms error is indicated in the following table

Bilinear interpolation		Biquadratic interpolation	
Number of elements,N	rms error	Number of elements,N	rms error
4	0.1537		
16	0.0418	4	0.0137
64	0.0106	16	0.00170
		64	0.000212

It can be seen that bilinear interpolation is converging like Δx^2 and biquadratic interpolation like Δx^3. This behaviour and the accuracy is similar to the results shown in Table 5.7.

5.10 Using (5.70) and Fig. 5.17 we obtain $d\xi = (2/\Delta x)dx$

For element j : $\phi_{j-1} = 0.5(1 - \xi)$, $\phi_j = 0.5(1 + \xi)$
$\partial\phi_{j-1}/\partial\xi = -0.5$, $\partial\phi_j/\partial\xi = 0.5$

For element $j+1$: $\phi_j = 0.5(1-\xi)$, $\phi_{j+1} = 0.5(1+\xi)$
$\partial\phi_j/\partial\xi = -0.5$, $\partial\phi_{j+1}/\partial\xi = 0.5$

Thus $M_{x1} = \frac{1}{\Delta x}\frac{\Delta x}{2}\int_{-1}^{1} 0.25(1-\xi)(1+\xi)d\xi = 1/6$

$$M_{x2} = 0.5\left[\int_{-1}^{1} 0.25(1+\xi)^2 d\xi + \int_{-1}^{1} 0.25(1-\xi)^2 d\xi\right] = 2/3$$

$$M_{x3} = 0.5\int_{-1}^{1} 0.25(1-\xi)(1+\xi)d\xi\,] = 1/6$$

$$L_{x1} = \frac{1}{\Delta x}\int_{-1}^{1}(-0.5)0.5(1+\xi)d\xi = -0.5/\Delta x$$

$$L_{x2} = \frac{1}{\Delta x}\left[\int_{-1}^{1} 0.25(1+\xi)d\xi + \int_{-1}^{1}(-0.25)(1-\xi)d\xi\right] = 0$$

$$L_{x3} = \frac{1}{\Delta x}\int_{-1}^{1}(0.5)0.5(1+\xi)d\xi = 0.5/\Delta x$$

$$L_{xx1} = -\frac{2}{\Delta x^2}\int_{-1}^{1}(-0.5)0.5d\xi = 1/\Delta x^2$$

$$L_{xx2} = -\frac{2}{\Delta x^2}\left[\int_{-1}^{1} 0.25 d\xi + \int_{-1}^{1} 0.25\ d\xi\right] = -2/\Delta x^2$$

$$L_{xx3} = -\frac{2}{\Delta x^2}\int_{-1}^{1} -0.25\ d\xi = 1/\Delta x^2\ .$$

5.11 Equation (5.73), applied at $m = J$, gives the following for (5.77),

$$b_{J,J} = b/\Delta x_J - 1/\Delta x_J + \Delta x_J/3$$

for linear interpolation, and

$$b_{J,J} = b/\Delta x_J - 7/(6\Delta x_J) + 4\Delta x_J/15\ ,$$

for quadratic interpolation. However the additional terms are combined with g_J. Similarly the term $b_{2,1}a$ is subtracted from g_1 and, for quadratic interpolation, $b_{1,2}$a is subtracted from g_2. The accuracy of the resulting solution is indicated in the table.

Δx	rms error (linear)	rms error (quadratic)
1/8	0.00414	0.000455
1/12	0.00181	0.000088
1/16	0.00101	0.000025
1/20	0.00064	0.000012

The results indicate accuracies comparable to those in Table 5.10 and that the theoretical convergence rates are being achieved.

5.12 Application of a linear finite element analysis to $dy/dx - y = 0$ provides the following coefficients, equivalent to (5.79),

$$b_{j,j-1} = -0.5 - \Delta x/6,\quad b_{j,j} = -2\Delta x/3,\quad b_{j,j+1} = 0.5 - \Delta x/6.$$

For $j = J$, $b_{J,J-1} = -0.5 - \Delta x/6$, $b_{J,J} = -\Delta x/3 + 0.5$.

For quadratic interpolation and Galerkin corner nodes,

$$b_{j,j-2} = 1/6 + \Delta x/15,\quad b_{j,j-1} = -(2/3 + 2\Delta x/15),\quad b_{j,j} = -8\Delta x/15$$

$$b_{j,j+1} = 2/3 - 2\Delta x/15,\quad b_{j,j+2} = -1/6 + \Delta x/15.$$

For $j = J$, $b_{J,J} = -4\Delta x/15 + 0.5\Delta x$.

For quadratic interpolation and midside Galerkin nodes,

$$b_{j,j-1} = -(2/3 + 2\Delta x/15),\quad b_{j,j} = -16\Delta x/15,\quad b_{j,j+1} = 2/3 - 2\Delta x/15\ .$$

For equation formed at $j = 2$ and 3 (quadratic interpolation) adjustment is required for the boundary condition, $y(0) = 1.0$. The solution accuracy is indicated in the table

NX	linear FEM rms error	quadratic FEM rms error
5	0.0405	0.0161
9	0.0100	0.00377
21	0.00158	0.000579
41	0.00040	0.00014

The traditional Galerkin method produces an rms error = 0.00046 with 3 degrees of freedom (Table 5.3). Linear FEM requires about 40 degrees of freedom and quadratic FEM requires about 20 degrees of freedom to achieve comparable accuracy. However the FEM stiffness matrix is sparse so that it is more economical per degree of freedom than the traditional Galerkin method. This leads to a greater computational efficiency for problems with non-smooth solutions requiring a large number of degrees of freedom.

5.13 Program DUCT provides a uniform grid in both directions. Consequently for small values of b/a and an equal number of grid points in both directions the grid aspect ratio becomes large and the grid is very coarse next to $x = \pm a$. Attempts to obtain a solution approximating a two-dimensional profile using an 11×11 grid produce rather poor solutions for $b/a < 0.2$. Typical solution accuracies for decreasing b/a produced by a linear finite element method on an 11×11 grid are shown in the following table.

b/a	rms_{1D} error	rms_{2D} error	$w_{c/l}/\bar{w}$
1.00	0.491	0.00036	2.11
0.50	0.382	0.00064	2.00
0.20	0.169	0.00187	1.73
0.10	0.087	0.00443	1.62
0.05	0.053	0.00882	1.57
0.02	0.041	0.01190	1.56

The term, rms_{1D} error, is an rms error for the solution across the narrow section, $-b < y < b$, at $x = 0$. The term, rms_{2D} error, is the rms error for the complete domain. With decreasing b/a the centre-line solution, at $x = 0$, is gradually approaching a one-dimensional parabolic profile. However for $b/a = 0.02$, the solution close to $x = \pm a$ is oscillatory which causes the large value of rms_{2D} error. It is also clear that the limiting value of $w_{c/l}/\bar{w}$ is greater than 1.5. Thus to get more accurate results, for small b/a, a finer grid is required in the x direction.

5.14 Evaluation of the Galerkin FEM leads to $\underline{\mathbf{B}}\ \mathbf{W} = 0$ in place of (5.100), where $b_{m,i}$ is given by (5.101) with $b/a = 1.0$. Equation (5.101) can be split into the product of integrals in the x and y directions as in Appendix A.2.

For ϕ_m centered at $x = 0$ only the two right-hand elements contribute. Thus L_{yy} and M_y have the same evaluation as in the interior. L_{xx3} and M_{x3} have the same values as in the interior since they only receive contributions from the right-hand elements. There are no contributions to L_{xx2} and M_{x2} from the left-hand elements so the boundary values are half the interior values for a uniform mesh.

To modify DUCT to obtain numerical values it is necessary to:

a) replace line 29 with DX = 1./ANX
b) replace lines 42 & 45 with X = AJ*DX
& Y = AK*DY
c) replace line 47 with WD = COS(0.5*PI*X)*EXP(0.5*PI*Y)
d) replace line 63 with DO 12 J = 1, NXP
e) after line 65 insert IF(J.EQ.1)JM = JP
f) remove 1. from lines 70 & 74
g) change one ANX-1 to ANX in lines 81 and 103
h) replace line 92 with DO 18 J = 1, NXP

Item e) is a convenient way to implement the homogeneous Neumann boundary condition at x=0 when the grid is uniform. This technique is discussed at the end of Sect. 8.4.2. The results (see table) indicate second-order convergence and a similar accuracy for both linear finite elements and a three-point centered difference scheme.

grid	finite element rms error	finite difference rms error
6x6	0.00608	0.00592
11x11	0.00136	0.00135
21x21	0.00030	0.00027

5.15 The following terms appear in the discrete equation

$$M_y \otimes (L_x - \alpha L_{xx})\phi + M_x \otimes (L_y - \beta L_{yy})\phi = 0 .$$

For the case $\alpha = 1.0$, $\beta = 1.0$ the following changes are needed to DUCT:

a) replace line 29 with DX = 1./ANX
b) after line 36 insert PAR4 = 1./12./DX
PAR5 = 1./12./DY
c) replace line 47 with WD = EXP(X)*EXP(Y)
d) after line 71 insert

DUM = DUM - (PAR4+PAR5)*(WA(JP,KP)-WA(JM,KM))
DUM = DUM - (PAR4-PAR5)*(WA(JP,KM)-WA(JM,KP))
DUM = DUM - 4.*PAR4*(WA(JP,K) - WA(JM,K))
DUM = DUM - 4.*PAR5*(WA(J,KP) - WA(J,KM))

e) after line 74 insert

DUM = DUM + 0.5*(WA(JM,K) - WA(JP,K))/DX+0.5*(WA(J,KM)-WA(J,KP))/DY

The resulting solutions demonstrate that both the centered difference three-point scheme and the linear finite element scheme are achieving comparable accuracy and second-order convergence, as indicated in the table.

grid	finite element rms error	finite difference rms error
6x6	0.00100	0.00093
11x11	0.00022	0.00021
21x21	0.000037	0.000031

5.16 Discussion of the tabulated results is provided below.

t	Approx. J = 5	Solution, J = 7	T J = 9	Approx. Solution, T_b	Exact Solution, $\bar{T}$
0.00	1.5000	1.5000	1.5000	1.5000	1.5000
0.02	1.3383	1.3414	1.3399	1.3404	1.3408
0.04	1.1911	1.1941	1.1926	1.1931	1.1943
0.06	1.0669	1.0700	1.0685	1.0690	1.0707
0.08	0.9647	0.9677	0.9662	0.9667	0.9686
0.10	0.8807	0.8837	0.8823	0.8828	0.8846
0.12	0.8118	0.8148	0.8134	0.8139	0.8157
0.14	0.7553	0.7583	0.7569	0.7574	0.7592
0.16	0.7090	0.7120	0.7106	0.7111	0.7128
0.18	0.6710	0.6740	0.6726	0.6731	0.6746
0.20	0.6399	0.6429	0.6415	0.6420	0.6434

Application of the Galerkin spectral method produces ordinary differential equations,

$$da_m/dt + \alpha(m\pi)^2 a_m + 32\alpha/(m\pi) = 0, \quad m = 1, 3, 5 \text{ etc.}$$

and $\quad a_m = 0, \quad m = 2, 4, 6$ etc.

Marching this equation in time using (5.130) and comparing the solution given by the equivalent of (5.123) with (5.131) and (5.132) gives the solutions shown in the table. It can be seen, that with increasing J, the approximate solution is gradually converging to the solution T_b, as discussed in the text.

5.17 A solution of the following form is assumed

$$T(x,t) = 3 - 2x + b_o + \sum_{j=1}^{J}[a_j \sin(j\, 2\pi x) + b_j \cos(j\, 2\pi x)].$$

In order to satisfy $\partial T/\partial x = -2$ at $x = 0$ and $T = 1$ at $x = 1$,

we choose $b_o = -\sum_{j=1}^{J} b_j;$ and $a_J = -\sum_{j=1}^{J-1} a_j(j/J)$.

The starting values of a_j and b_j are chosen to match the initial condition, $T(x,o) = 3 - 2x - 2x^2 + 2x^3$. This leads to $a_j(o) = 3/(j\pi)^3, \quad b_j(o) = 1/(j\pi)^2$.

Using the tau method the solutions for $a_j(t)$ and $b_j(t)$ are given by (5.146). The solution with $J = 11$ at $t = 0.10$ is used as an exact solution. The rms errors at $t = 0.10$ for $J = 3$, 4 and 5 imply convergence and a fairly high accuracy, as indicated in the table.

J	rms error (NX=11) for t = 0.10
3	0.00046
4	0.00036
5	0.00012

It may be noted that for $J = 11$ the solution is already fairly close to the steady-state value at $t = 0.10$.

CTFD Solutions Manual: Chapter 6

Steady Problems

6.1 For this problem it is useful to consider the case where (6.9) are repeated ten times making an overall system of 30 equations. The enlarged system converges in the same number of iterations as the original problem since all blocks are independent but it produces a more realistic operation count for FACT and SOLVE. For a starting solution of $T_o = (0.80, 0.50, 0.10, 0.80 \ldots\ldots)$ the results are summarised in the table.

p	Iterations to Convergence	Calls to JACOB and FACT	Total Operation Count
1	5	5	78600
2	7	4	69090
3	8	3	57510
5	11	3	63720
6	11	2	50070
8	12	2	52140
10	13	2	54210

Subroutines JACOB and FACT are called on the first iteration and then every p iterations. The operation counts are based only on floating point operations and assume that all operations require the same execution time. The operation counts per iteration are

RESID = 270, JACOB = 150, FACT = 13500, SOLVE = 1800 .

It is clear that there is an advantage in evaluating JACOB and FACT less often. Since FACT is an N^3 process the benefit will increase with N, as long as JACOB is dense. For this example JACOB is tridiagonal, so for large N it would be better to replace FACT and SOLVE with BANFAC and BANSOL.

6.2 For Re = 1, 2, 5 and 10, and DT = 5000, the optimum ω to obtain convergence is indicated in the following table, as well as the number of iterations, N_{it}

Re	1	2	5	10	10	10	10
DT	5000	5000	5000	5000	0.1	0.01	0.001
ω_{opt}	0.26	0.26	0.26	0.26	0.42	1.28	4.92
N_{it}	95	86	74	65	42	17	22

For the large value, DT = 5000, NEWTBU is behaving like a conventional Newton's method. The optimum $\omega = 0.26$. Using a larger value of ω causes a rapid increase in the number of iterations to convergence and eventually divergence. Smaller values of ω cause the number of iterations to grow slowly. The number of iterations to convergence reduces with increasing Re due to the more severe solution gradient and the concentration of large residuals in that area.
Reducing DT increases the diagonal dominance and allows the iteration to remain stable with a larger value of ω. Consequently the number of iterations to convergence is reduced. However reducing DT also causes a departure from the true Jacobian; eventually (small DT) the number of iterations begins to rise again.

6.3 The best choice, ω_m, is obtained by assuming that R_{rms} is a quadratic function of ω. Therefore $R_{rms}^{(m)}$ is evaluated for m = 1,3 where $\omega_m = \omega^o - \Delta\omega$, ω^o, $\omega^o + \Delta\omega$. Here $\omega^o = \omega_{opt}^{(n)}$ and $\Delta\omega$ is chosen empirically. For a quadratic function,

$\omega_{opt}^{(n+1)} = \omega^o - ac/bc$

where $ac = 0.5(R_{rms}^{(3)} - R_{rms}^{(1)})/\Delta\omega$

and $bc = (R_{rms}^{(1)} - 2R_{rms}^{(2)} + R_{rms}^{(3)})/\Delta\omega^2$.

The required coding is shown below and inserted after line 111 of NEWTBU. Tests are included to check that $\omega_1 < \omega_{opt} < \omega_3$. BPS provides an asymmetric weighting which biases the value of $\omega_{opt}^{(n+1)}$ to lower values. The choice BPS = 0.65 usually produces a stable convergence pattern.

```
C
C     QUADRATIC OM SEARCH
C
      OML(1) = OM - DOM
      OML(2) = OM
      OML(3) = OM + DOM
      DO 204 L = 1,3
      DO 202 J = 2,NXP
      DO 201 K = 2,NYP
      MB = 2*(K-2) + 2*(NY-2)*(J-2)
      DU = - RD(MB+1)
      DV = - RD(MB+2)
      UL(K,J) = U(K,J) + DU*OML(L)
      VL(K,J) = V(K,J) + DV*OML(L)
  201 CONTINUE
  202 CONTINUE
C
      CALL RESBU(UL,VL,R)
C
      SUM = 0.
      DO 203 I = 1,N
  203 SUM = SUM + R(I)*R(I)
      RMSL(L) = DSQRT(SUM/AN)
  204 CONTINUE
C
      AC = 0.5*DMI*(RMSL(3)-RMSL(1))
      BC = DMS*(RMSL(1)-2.*RMSL(2)+RMSL(3))
      IF(BC .GT. 1.0E-07)GOTO 206
      IF(AC .GT. 1.0E-07)OM = BPS*OML(1)+(1.0-BPS)*OML(2)
      IF(AC .LT. -1.0E-07)OM = BPS*OML(2)+(1.0-BPS)*OML(3)
      GOTO 207
  206 OM = OML(2) - AC/BC
      IF(OM .LT. OML(1))OM = BPS*OML(1)+(1.0-BPS)*OML(2)
      IF(OM .GT. OML(3))OM = BPS*OML(2)+(1.0-BPS)*OML(3)
  207 CONTINUE
```

Typical results for Re = 10 and various values of DT are shown in the following table

DT	5000	0.10	0.01
ω_{st}	0.20	0.20	0.70
$\Delta\omega$	0.02	0.02	0.05
BPS	0.70	0.60	0.60
N_{it}	64	46	26

The algorithm is converging in slightly more iterations than if a single empirically chosen value of ω is used (Problem 6.2), depending on the choice of ω_{st}. However the algorithm is quite robust particularly if combined with small values of DT. The major overhead is the three additional evaluations of RESBU per iteration.

6.4 The required coding for this problem is provided by Program TRIDA. It may be noted that VEX is the exact solution of the discrete problem. Not surprisingly this solution is reproduced in file TRIDA.OUT.

```
      Program TRIDA
C
C     USE BANFAC AND BANSOL TO SOLVE A TRIDIAGONAL SYSTEM
C
      DOUBLE PRECISION B(5,65),G(65),V(65),VEX(65)
C
      OPEN(1,FILE='TRIDA.DAT')
      OPEN(6,FILE='TRIDA.OUT')
      READ(1,1)N,RCELL
    1 FORMAT(I5,F5.2)
      WRITE(6,2)N,RCELL
    2 FORMAT(' TRIDIAGONAL PROBLEM,  N=',I2,'  RCELL=',F5.2)
C
C     SET VEX
C
      CON = 5.0/3.0
      DO 3 J = 1,N
    3 VEX(J) = -0.0060834 + 0.00365*(CON)**J
C
C     SET B AND G
C
      NL = N-2
      NLP = N-1
      ANL = NL
      DO 4 J = 1,NL
      B(1,J) = 0.0
      B(2,J) = -(1.0 + 0.5*RCELL)
      B(3,J) = 2.0
      B(4,J) = -(1.0 - 0.5*RCELL)
      B(5,J) = 0.0
      G(J) = 0.
    4 CONTINUE
      G(1) =  - B(2,1)*VEX(1)
      B(2,1) = 0.
      G(NL) =  - B(4,NL)*VEX(N)
      B(4,NL) = 0. C
      CALL BANFAC(B,NL,1)
C
      CALL BANSOL(G,V,B,NL,1)
C
C     COMPUTE ERROR
C
```

```
      SUM = 0.
      DO 5 J = 1,NL
      JA = J+1
      DIF = V(J) - VEX(JA)
    5 SUM = SUM + DIF*DIF
      RMS = SQRT(SUM/ANL)
      WRITE(6,6)(V(J),J=1,NL)
      WRITE(6,7)(VEX(J),J=2,NLP)
      WRITE(6,8)N,RCELL,RMS
    6 FORMAT('  V=',11F7.3)
    7 FORMAT(' VX=',11F7.3)
    8 FORMAT(' N=',I2,' RCELL=',E10.3,' RMS=',E10.3)
      STOP
      END
```

TRIDA.OUT

```
 TRIDIAGONAL PROBLEM,    N=11    RCELL= 0.50
  V= 0.004 0.011 0.022 0.041 0.072 0.124 0.211 0.356 0.598
 VX= 0.004 0.011 0.022 0.041 0.072 0.124 0.211 0.356 0.598
 N=11   RCELL= 0.500E+00   RMS= 0.998E-08
```

6.5 The following modifications, with an appropriate test on INT, are required to execute a pentadiagonal system in subroutines BANFAC and BANSOL.

Modifications to Subroutine BANFAC

```
C
C     TWO PASSES REQUIRED FOR A PENTADIAGONAL SYSTEM
C
    6 DO 7 J = 2,NP
      JP = J+1
      B(1,JP) = B(1,JP)/B(2,J)
      B(2,JP) = B(2,JP) - B(1,JP)*B(3,J)
      B(3,JP) = B(3,JP) - B(1,JP)*B(4,J)
      IF(JP .GE. N)GOTO 7
      B(4,JP) = B(4,JP) - B(1,JP)*B(5,J)
    7 CONTINUE
      DO 8 J = 1,NP
      JP = J+1
      B(2,JP) = B(2,JP)/B(3,J)
      B(3,JP) = B(3,JP) - B(2,JP)*B(4,J)
      IF(JP .GE. N)GOTO 8
      B(4,JP) = B(4,JP) - B(2,JP)*B(5,J)
    8 CONTINUE
      RETURN
```

Modifications to Subroutine BANSOL

```
C     TWO PASSES REQUIRED FOR PENTADIAGONAL SYSTEM
C
    8 NP = N-1
      DO 9 J = 2,NP
      JP = J+1
    9 R(JP) = R(JP) - B(1,JP)*R(J)
      DO 10 J = 1,NP
      JP = J+1
   10 R(JP) = R(JP) - B(2,JP)*R(J)
      X(N) = R(N)/B(3,N)
      X(N-1) = (R(N-1) - B(4,N-1)*X(N))/B(3,N-1)
      DO 11 J = 2,NP
      JA = N-J
      X(JA) = (R(JA) - B(4,JA)*X(JA+1) - B(5,JA)*X(JA+2))/B(3,JA)
   11 CONTINUE
      RETURN
```

For R_{cell} = 1.0 and 11 grid points the solution is indicated in the following table.

x	0.2	0.3	0.4	0.5	0.6	0.7	0.8
v	0.000	0.001	0.003	0.007	0.020	0.052	0.138
v_{ex}	0.000	0.001	0.002	0.007	0.018	0.050	0.135

For x = 0, 0.1, 0.9 and 1.0, v is chosen to coincide with v_{ex}. The rms error for the above solution is 0.0016 and is dominated by the solution close to x = 1.0, at which v = v_{ex} = 1.0.

6.6 After discretisation of (6.104), the non-zero coefficients of $\underline{a}$, $\underline{b}$ and $\underline{c}$ in (6.36) are

$$a_{i,1,1} = \Delta x, \quad a_{i,1,2} = 1.0, \quad a_{i,2,1} = 1.0, \quad a_{i,2,2} = 0.25\ \Delta x$$

$$b_{i,1,2} = -2.0, \quad b_{i,2,1} = -2.0$$

$$c_{i,1,1} = -\Delta x, \quad c_{i,1,2} = 1.0, \quad c_{i,2,1} = 1.0, \quad c_{i,2,2} = -0.25\ \Delta x$$

Since Dirichlet boundary conditions are required, values of d(1) and d(n-2) are set from $\underline{a}_1$, $\underline{c}_{n-2}$ and the boundary conditions. A listing of subroutine BLTRI is provided. A typical solution for n = 11 is shown below. Since T and S are smoothly varying the rms errors are quite small.

```
      Subroutine BLTRI(N,A,B,C,D,X)
C
C      SOLVES AN NBL x NBL BLOCK TRIDIAGONAL SYSTEM OF
C      EQUATIONS, P6.6
C
      PARAMETER(NBL=2)
      DIMENSION A(50,5,5),B(50,5,5),C(50,5,5),D(50,5),DD(5)
      DIMENSION X(50,5),CD(50,5,6),BD(50,50),JPVT(50),R(50)
C
      NBP = NBL + 1
      DO 2 K = 1,NBP
      DO 1 J = 1,NBL
    1 CD(1,J,K) = 0.
    2 CONTINUE
C
C      FORWARD SWEEP
C
      DO 9 L = 1,N
      DO 5 J = 1,NBL
      DD(J) = D(L,J)
      DO 4 K = 1,NBL
      BD(J,K) = B(L,J,K)
      DO 3 I = 1,NBL
      IF(K .EQ. NBL)DD(J) = DD(J) - A(L,J,I)*CD(L,I,NBP)
    3 BD(J,K) = BD(J,K) - A(L,J,I)*CD(L,I,K)
    4 CONTINUE
    5 CONTINUE
C
      CALL FACT(NBL,BD,JPVT)
C
      KST = 1
      IF(L .EQ. N)KST = NBP
      DO 8 K = KST,NBP
      DO 6 J = 1,NBL
      IF(K .LE. NBL)R(J) = C(L,J,K)
      IF(K .EQ. NBP)R(J) = DD(J)
    6 CONTINUE
C
      CALL SOLVE(NBL,BD,JPVT,R)
C
      DO 7 J = 1,NBL
    7 CD(L+1,J,K) = R(J)
    8 CONTINUE
    9 CONTINUE
C
C      BACKWARD SWEEP
C
      DO 10 J = 1,NBL
   10 X(N,J) = CD(N+1,J,NBP)
      DO 13 L = 2,N
      LA = N + 1 - L
      DO 12 J = 1,NBL
      X(LA,J) = CD(LA+1,J,NBP)
      DO 11 K = 1,NBL
   11 X(LA,J) = X(LA,J) - CD(LA+1,J,K)*X(LA+1,K)
   12 CONTINUE
   13 CONTINUE
      RETURN
      END
```

Typical solution

```
BLOCK TRIDIAGONAL PROBLEM   2 x 2   N=11
  S=0.1000E+01 0.1005E+01 0.1020E+01 0.1045E+01 0.1081E+01 0.1128E+01
    0.1185E+01 0.1255E+01 0.1337E+01 0.1433E+01 0.1543E+01
SEX=0.1000E+01 0.1005E+01 0.1020E+01 0.1045E+01 0.1081E+01 0.1128E+01
    0.1185E+01 0.1255E+01 0.1337E+01 0.1433E+01 0.1543E+01
  T=0.0000E+00 0.2003E+00 0.4026E+00 0.6090E+00 0.8214E+00 0.1042E+01
    0.1273E+01 0.1517E+01 0.1776E+01 0.2053E+01 0.2350E+01
TEX=0.0000E+00 0.2003E+00 0.4027E+00 0.6090E+00 0.8215E+00 0.1042E+01
    0.1273E+01 0.1517E+01 0.1776E+01 0.2053E+01 0.2350E+01
NBL,N= 211   RMSS,RMST= 0.8957E-04 0.8578E-04
```

6.7 To implement Jacobi iteration it is necessary to introduce an auxiliary array, WAJ(41,41), in line 78 and to include nested do loops setting WA(J,K) = WAJ(J,K) after line 82. The results for an 11×11 grid and TOL = 1.0×10^{-6} are indicated in the table.

b/a	1.0	3.0	10.0
λ	N_{it}	N_{it}	N_{it}
Jacobi	87	54	20
1.0 (G.S.)	51	35	17
1.55 (SOR)	15	12	8

As expected the SOR technique produces convergence in the fewest iterations. Also as b/a increases the problem becomes more one-dimensional and consequently fewer iterations are needed to reach convergence.

6.8 The finite difference ADI scheme for the DUCT problem (Sect. 5.5.2) is implemented as (6.66 and 6.67). The finite element ADI scheme is obtained by retaining only the j line contributions of $(b/a)^2\partial^2 w/\partial x^2$ on the implicit side during the first half step. This corresponds to $M_{y2}\otimes L_{xx}w_{j,k}$ in (5.106). The terms $M_{y1}\otimes L_{xx}w_{j,k}$ and $M_{y3}\otimes L_{xx}w_{j,k}$ are treated explicitly during the first half step. In an equivalent way $M_{x2}\otimes L_{yy}w_{j,k}$ is treated implicitly in the second half step.

The following additional arrays are required

DIMENSION WB(41,41), $B(5,65)$, RHS(65), WAD(65), EM(3)

WB is upgraded in the first half step to avoid overwriting WA. B, RHS and WAD are needed for BANFAC/BANSOL. The parameter ALY is read in. The following code to set parameters is introduced after line 36.

```
      ALX = ALY*(BAR*DY/DX)**2
      ALYH = ALY
      EM(1) = 1./6.
      IF(ME .EQ. 2)EM(1) = 0.
      EM(3) = EM(1)
      EM(2) = 1.0 - EM(1) - EM(3)
      DO 52 L = 1,65
      DO 51 K = 1,65
   51 B(K,L) = 0.
   52 CONTINUE
```

The SOR iterative scheme, lines 58-83, is replaced with the following ADI code.

```
C     ITERATE USING ADI
C
      LCT = 0
      LCMAX = 10
      ALMAX = LCMAX - 1
      DO 13 I = 1,ITMX
      LCT = LCT + 1
      IF(LCT .GE. LCMAX)LCT = 1
      ALC = LCT
      ANG = 0.5*PI*ALC/ALMAX
      ALY = ALYH*(0.5/SIN(ANG))**2
      ALX = ALY*(BAR*DY/DX)**2
C
C     TRIDIAG IN X-DIRECTION
C
      DO 108 K = 2,NYP
      KM = K-1
      KP = K+1
      DO 10 J =2,NXP
      JM = J-1
```

```
      JP = J+1
      B(2,JM) = -ALX*EM(2)
      B(3,JM) =  2.0*ALX*EM(2) + 1.0
      B(4,JM) = -ALX*EM(2)
      IF(ME .EQ. 2)GOTO 9
      RHS(JM) = WA(J,K) + ALY*DYS + (ALY+ALX)*(WA(JM,KM)
     1+WA(JM,KP)+WA(JP,KM)+WA(JP,KP))/6.+(2.0*ALY-ALX)
     2*(WA(J,KM)+WA(J,KP))/3.
     3-ALY*(WA(JM,K)+4.0*WA(J,K)+WA(JP,K))/3.
      GOTO 10
    9 RHS(JM) = ALY*DYS + ALY*(WA(J,KM)+WA(J,KP))
     1 +(1.0-2.*ALY)*WA(J,K)
   10 CONTINUE
      RHS(1) = RHS(1) - B(2,1)*WA(1,K)
      RHS(JM) = RHS(JM) - B(4,JM)*WA(NX,K)
      B(2,1) = 0.
      B(4,JM) = 0.
C
      CALL BANFAC(B,JM,1)
C
      DO 105 J = 2,NXP
      JM = J-1
  105 WB(J,K) = WAD(JM)
  108 CONTINUE
C
C     TRIDIAG IN Y-DIRECTION
C
      DO 118 J = 2,NXP
      JM = J-1
      JP = J+1
      DO 114 K = 2,NYP
      KM = K-1
      KP = K+1
      B(2,KM) = -ALY*EM(2)
      B(3,KM) =  2.0*ALY*EM(2) + 1.0
      B(4,KM) = -ALY*EM(2)
      IF(ME .EQ. 2)GOTO 112
      RHS(KM) = WB(J,K) + ALY*DYS + (ALY+ALX)*(WB(JM,KM)
     1 +WB(JM,KP)+WB(JP,KM)+WB(JP,KP))/6.+(2.0*ALX-ALY)
     2 *(WB(JM,K)+WB(JP,K))/3.
     3-ALX*(WB(J,KM)+4.0*WB(J,K)+WB(J,KP))/3.
      GOTO 114
  112 RHS(KM) = ALY*DYS+ ALX*(WB(JM,K)+WB(JP,K))
     1 +(1.0-2.*ALX)*WB(J,K)
  114 CONTINUE
      RHS(1) = RHS(1) - B(2,1)*WB(J,1)
      RHS(KM) = RHS(KM) - B(4,KM)*WB(J,NY)
      B(2,1) = 0.
      B(4,KM) = 0.
C
      CALL BANFAC(B,KM,1)
C
      DO 116 K = 2,NYP
      KM = K-1
      SUM = SUM + (WA(J,K) - WAD(KM))**2
  116 WA(J,K) = WAD(KM)
  118 CONTINUE
C
      RMS = SQRT(SUM/(ANX-1.)/(ANY-1.))
      WRITE(*,119)I,RMS
  119 FORMAT(' I,RMS=',I5, E10.3)
      IF(RMS .LE. EPS)GOTO 15
   13 CONTINUE
```

This code includes the sequence of iteration parameters (but with ALY, ALX commented out),

$\lambda_y = \lambda_{yh}\{0.5/[\sin(0.5\pi\ell/N)]\}^2$.

The number of iterations to convergence, equivalent to Table 6.3, but with TOL = 1.0×10^{-7}, are shown below

λ_y	1.0	1.5	2.9	$\lambda_{yh} = 0.9$	$\lambda_{yh} = 1.2$
ADI-FEM	23	16	div.	7	11
ADI-FDM	23	17	11	13	6

The ADI-FEM scheme, with fixed λ_y, is not able to maintain a stable iteration for $\lambda_y \geq 1.6$; the lowest value of 16 iterations occurs at $\lambda_y = 1.5$. The lowest value of 11 iterations at $\lambda_y = 2.9$ is achieved for ADI-FDM; stable results are obtained for $\lambda_y = 3.9$. The number of iterations is reduced by using a sequence of λ_y values. The two results shown for $\lambda_{yh} = 0.9$ and 1.2 are optimal for the ADI-FEM and ADI-FDM schemes respectively.

6.9 To fit in with the multigrid formulation, the single-grid SOR algorithm, (6.63 and 6.64), is rewritten as

$$RES^{(n)} = RHS + \{(b/a)^2 L_{xx} + L_{yy}\} w_{j,k}^{(n)} \qquad (1)$$

and

$$w_{j,k}^{(n+1)} = w_{j,k}^{(n)} + (0.5\lambda/PAR1) RES^{(n)} \ . \qquad (2)$$

For the single-grid algorithm, RHS = 1.0 . For the V-cycle multigrid method the above algorithm is used on the finest grid, and implemented in subroutine RELAX. After $\nu 1$ relaxations, the residual, $\mathrm{RES}_m^{(n+1,\nu 1)}$ is injected to the next coarser grid as

$RHS_{m-1} = I_m^{m-1} RES_m^{(n+1,\nu 1)}$.

Consequently the same algorithm, (1) and (2), can be used on all coarser grids, for both legs of the V-cycle. The code to implement one V-cycle, including calls to RELAX and both injection and interpolation, is provided in subroutine MUGDU. The current solutions for w and RHS on all grids are stored in one-dimensional arrays, Q and F, respectively. Typical results are indicated in the table for a 33 × 33 grid. The coarsest multigrid is 5 × 5 with two relaxations moving to a coarser grid and one moving to a finer grid.

Case	λ	CUR	N_{it}	$N_{it,eq}$
FDM, s.g.	1.8	–	84	84
FEM, s.g.	1.8	–	69	69
FDM, mg	1.5	0.5	51	80
FEM, mg	1.4	0.5	39	61

In interpolating to the finer grid, the coarse-grid solution correction is under-relaxed by the factor, CUR (subroutine MUGDU). Each multigrid cycle requires some extra work, therefore N_{it} indicates the actual number of calls to RELAX on the finest (33×33) grid, and $N_{it,eq}$ indicates the equivalent number of fine grid calls. For this particular example the single-grid algorithm is able to support a relatively large value of λ. Consequently multigrid does not provide a major improvement in computational efficiency. However the advantages of using multigrid would be greater for a finer grid, a lower residual tolerance or a less diagonally-dominant problem.

To implement multigrid, lines 58 to 83 of DUCT are replaced with the following code and CUR, LCOR, LFIN, LITD and LITU must be read in.

```
C     GENERATE MULTIGRID PARAMETERS INITIAL CONDITIONS
C
      LST = 1
      DO 82 L = 1,LFIN
      NA = 2**L + 1
      KS(L) = LST
   82 LST = LST + NA*NA
      DO 83 L = 1,LST
      Q(L) = 0.
   83 F(L) = 0.
      KJST = KS(LFIN) - 1
      DO 85 J = 1,NX
      JM = J-1
      DO 84 K = 1,NY
      KJ = NY*JM + K + KJST
      F(KJ) = 1.0
   84 Q(KJ) = WA(J,K)
   85 CONTINUE
C
C     ITERATE USING MULTIGRID (SOR FOR RELAXATION)
C
      DO 13 I = 1,ITMX
C
      CALL MUGDU(LCOR,LFIN,LITD,LITU,KS,Q,F,BAS,OM,CUR,RMS,ME)
C
      IF(RMS .LT. EPS)GOTO 15
   13 CONTINUE
```

Additional subroutines, MUGDU and RELAX, are required.

```
      Subroutine MUGDU(LCOR,LFIN,LITD,LITU,KS,Q,F,
     1BAS,OM,CUR,RMS,ME)
C
C     MULTIGRID  FOR DUCT PROBLEM
C
      DOUBLE PRECISION Q,F,RES
      DIMENSION Q(1500),F(1500),KS(6),RES(33,33)
      LCOS = LCOR + 1
      IF(LCOS .GT. LFIN)LCOS = LFIN
      DO 20 L = LFIN,LCOS,-1
      NF = 2**L+1
      NC = 2**(L-1)+1
      LST = KS(L)
      DO 10 LSUB = 1,LITD
   10 CALL RELAX(RES,Q(LST),F(LST),NF,BAS,OM,RMS,ME)
      IF(LCOS .EQ. LFIN)GOTO 20
C     INJECT TO COARSER GRID
      NM = NC - 1
      KJST = KS(L-1) - 1
      DO 14 J = 2,NM
      J2 = 2*J - 1
      JM = J - 1
      DO 12 K = 2,NM
      K2 = 2*K - 1
      KJ = NC*JM + K + KJST
   12 F(KJ) = RES(K2,J2)
   14 CONTINUE
```

```
   20 CONTINUE
C
      DO 40 L = LCOR,LFIN
      NC = 2**L+1
      NF = 2**(L+1)+1
      LST = KS(L)
      LSTF = KS(L+1)
      DO 30 LSUB = 1,LITU
   30 CALL RELAX(RES,Q(LST),F(LST),NC,BAS,OM,RMS,ME)
      IF(L .EQ. LFIN)GOTO 40
C     INTERPOLATE TO A FINER GRID
      KJSTF  = LSTF - 1
      KJSTC  = LST - 1
      DO 34 J = 2,NC
      JM = J - 1
      J2 = 2*J - 1
      J2M = J2 - 1
      DO 32 K = 2,NC
      K2 = 2*K - 1
      KJC = NC*JM + K + KJSTC
      KJCJM = KJC - NC
      KJF = NF*J2M + K2 + KJSTF
      KJFJM = KJF - NF
      Q(KJF) = Q(KJF) + Q(KJC)*CUR
      Q(KJFJM) = Q(KJFJM) + 0.5*(Q(KJC)+Q(KJCJM))*CUR
      Q(KJF-1) = Q(KJF-1) + 0.5*(Q(KJC)+Q(KJC-1))*CUR
   32 Q(KJFJM-1)=Q(KJFJM-1)+0.25*(Q(KJC)+Q(KJC-1)
     1+ Q(KJCJM) + Q(KJCJM-1))*CUR
   34 CONTINUE
      DO 38 J = 1,NC
      JM = J - 1
      DO 36 K = 1,NC
      KJC = NC*JM + K + KJSTC
   36 Q(KJC) = 0.
   38 CONTINUE
   40 CONTINUE
      RETURN
      END

      Subroutine RELAX(RES,WA,RHS,NX,BAS,OM,RMS,ME)
C
C     MULTIGRID RELAXATION FOR DUCT PROBLEM
C
      DOUBLE PRECISION WA,RHS,RES,SUM
      DIMENSION WA(NX,NX),RHS(NX,NX),RES(33,33)
      NXP = NX-1
      NYP = NXP
      ANX = NXP
      DX = 2.0/ANX
      DXS = DX*DX
      PAR1 = BAS + 1.0/DXS
      PAR2 = (2./DXS - BAS)/3.
      PAR3 = (2.*BAS - 1./DXS)/3.
      OML = 0.5*OM/PAR1
      IF(ME .EQ. 1)OML = 1.5*OML
      SUM = 0.0D0
      DO 4 K = 2,NYP
      KM = K-1
      KP = K+1
      DO 3 J = 2,NXP
      JM = J-1
      JP = J+1
      IF(ME .EQ. 2)GOTO 1
      RES(J,K) = RHS(J,K)+PAR1/6.*(WA(JM,KP)+WA(JP,KP)+
     1WA(JM,KM)+WA(JP,KM)) + PAR2*(WA(J,KP) + WA(J,KM))  +
     2PAR3*(WA(JM,K) +WA(JP,K)) -4.*PAR1*WA(J,K)/3.
      GOTO 2
    1 RES(J,K) = RHS(J,K)+BAS*(WA(JM,K)-2.0*WA(J,K)+WA(JP,K))
     1+ (WA(J,KM)-2.0*WA(J,K)+WA(J,KP))/DXS
    2 SUM = SUM + RES(J,K)*RES(J,K)
      WA(J,K) = WA(J,K) + OML*RES(J,K)
    3 CONTINUE
    4 CONTINUE
      RMS = DSQRT(SUM/(ANX-1.0)/(ANX-1.0))
      RETURN
      END
```

6.10 Subroutines JACBU and FACT are called on the first iteration and then every p iterations. The operation counts are based only on floating point operations and assume that all operations require the same execution time. The operation counts per iteration for a 6×6 grid are

RESBU = 315, JACBU = 297, FACT = 3078, SOLVE = 648.

The number of iterations, N_{it}, number of calls to JACBU and FACT, N_{jac}, and the total operation count, associated with the above four subroutines, are indicated below for conditions corresponding to Fig. 6.24.

p	N_{it}	N_{jac}	Total Operation Count
1	23	23	99774
5	23	5	39024
10	23	3	32274
20	23	2	28899
30	23	1	25524

For this problem the Jacobian, subroutine JACBU, is relatively insensitive to the current solution if Δt is small. Therefore it is only necessary to form and invert the Jacobian once.

6.11 To provide a more useful example, DT = 0.5 and OM = 0.3 are used in place of the values indicated in Fig. 6.24. All non-diagonal terms in AJ, in JACBU, are set to zero if less than TOL*(DTI+2.*(CCX+CCY)) . The number of iterations, N_{it}, to convergence for various values of TOL are shown in the table.

TOL	0.10	0.30	0.45	0.49	0.50
N_{it}	58	58	58	62	72

A related strategy is to retain only tridiagonal terms in the x-direction for the "x-equation" and tridiagonal terms in the y-direction for the "y-equation". This allows BANFAC and BANSOL to be used instead of FACT and SOLVE in two 9×9 systems. If JACBU and BANFAC are implemented at every step the total operation count for 91 iterations is 68,796. If JACBU and BANFAC are implemented only for the first step, the total operation count for 91 operations is 37,206. These results may be compared with those in Problem 6.10.

6.12 For conditions corresponding to Fig. 6.24 the number of iterations to convergence, N_{it}, is indicated in the following table

$\Delta t^{(o)}$	r	p	N_{it}
0.01	1.0	1	23
0.005	1.3	10	19
0.003	1.5	10	17
0.005	1.5	6	18
0.003	1.8	6	18
0.015	2.0	6	22

The geometric ratio, r, has been progressively increased. For each value, $\Delta t^{(o)}$ and p have been modified empirically to obtain small values of N_{it}. The strategy is modestly successful. All results have been obtained with $\omega = 1.0$. The solution to Problem 6.2 suggests it may also be effective to vary ω.

CTFD Solutions Manual: Chapter 7

One-Dimensional Diffusion Equation

7.1 **a)** The given scheme is

$$\frac{1}{\Delta t}\Big\{(\gamma+1)(T_j^{n+1}-T_j^n)-\gamma(T_j^n-T_j^{n-1})\Big\} \\ -\alpha\Big\{(1-\beta)L_{xx}T_j^n+\beta L_{xx}T_j^{n-1}\Big\}=0\ , \tag{1}$$

where $L_{xx}T_j^n=(T_{j+1}^n-2T_j^n+T_{j-1}^n)/\Delta x^2$.

Expanding every term in (1) as a Taylor series about T_j^n, and ignoring terms containing derivatives like $\frac{\partial}{\partial t}(T_{x^4})$ and those of higher order, we have

$$(\gamma+1)\left(T_t+\frac{\Delta t}{2}T_{tt}+\frac{\Delta t^2}{6}T_{t^3}+\frac{\Delta t^3}{24}T_{t^4}+\ldots\right) \\ -\gamma\left(T_t-\frac{\Delta t}{2}T_{tt}+\frac{\Delta t^2}{6}T_{t^3}-\frac{\Delta t^3}{24}T_{t^4}+\ldots\right) \\ -\alpha\Big\{(1-\beta)(T_{xx}+\frac{\Delta x^2}{12}T_{x^4})+\beta(T_{xx}+\frac{\Delta x^2}{12}T_{x^4}-\Delta t\ T_{xxt})\Big\} \\ +O(H)=0\ . \tag{2}$$

Substituting for the derivatives as indicated in (4.12), we have

$$T_t-\alpha T_{xx}+(2\gamma+1)\frac{\Delta t}{2}\alpha^2\ T_{x^4} \\ -\alpha\left[T_{xx}+\frac{\Delta x^2}{12}T_{x^4}-\beta\ \Delta t\ \alpha\ T_{x^4}\right]+O(H)=0\ . \tag{3}$$

Thus $E_j^n=s\Delta x^2\alpha\ T_{x^4}\,[\gamma+(1/2)-(1/12s)+\beta]+O(H)$,

after simplification. This agrees with (7.14).

b) The algorithm is

$$T_j^{n+1}=\frac{1+2\gamma}{1+\gamma}T_j^n-\frac{\gamma}{1+\gamma}T_j^{n-1}+\frac{s}{1+\gamma}L'_{xx}\{(1-\beta)T_j^n+\beta T_j^{n-1}\} \tag{4}$$

with $L'_{xx}T_j=T_{j-1}-2T_j+T_{j+1}$.

The corresponding 'error equation' is

$$\begin{aligned}\xi_j^{n+1} &= \frac{1+2\gamma}{1+\gamma}\xi_j^n - \frac{\gamma}{1+\gamma}\xi_j^{n-1} \\ &+ \frac{s}{1+\gamma}\{(1-\beta)(\xi_{j-1}^n - 2\xi_j^n + \xi_{j+1}^n) + \beta(\xi_{j-1}^{n-1} - 2\xi_j^{n-1} + \xi_{j+1}^{n-1})\}\ .\end{aligned} \tag{5}$$

Upon substitution of $\xi_j^n = G^n e^{i\theta j}$ and simplification,

$$G = \frac{1+2\gamma}{1+\gamma} - \frac{\gamma}{1+\gamma}\frac{1}{G} + \frac{s}{1+\gamma}\left\{(1-\beta)(-2+2\cos\theta) + \frac{\beta}{G}(-2+2\cos\theta)\right\}$$

and

$$(1+\gamma)G^2 - G[1+2\gamma+2s(1-\beta)(\cos\theta-1)] + \gamma - 2\beta s(\cos\theta - 1) = 0\ ,$$

as required.

7.2 Operation counts are required for the FTCS, DuFort-Frankel and 3L-4TH schemes. Ignoring the operation counts in calculating the initial conditions and in imposing the boundary conditions, we have

$$\begin{aligned}OPCT_{FTCS} &= (7FX + 5FL + R)(N_x - 2)N_t \\ &+ (4FX + 3FL + 2R + 2L)(N_x - 2) \\ &= 5.45(N_x-2)N_t + 3.34(N_x - 2)\ .\end{aligned} \tag{1}$$

$$OPCT_{DuFort-Frankel} = 4.35\ N_t(N_x - 2) + 3.6(N_x - 2) + 9.55\ . \tag{2}$$

$$OPCT_{3L-4TH} = 22.2\ N_t(N_x-2) + 3.45(N_x - 2)\ . \tag{3}$$

When $s = 0.3$, each of the schemes takes 24 time steps to reach a time level of 9 secs, with $N_x = 11$.

The corresponding operation counts are 1210, 930 and 4850.

The computational efficiency is given by $CE = k/(\epsilon\ CPT)$, where,

$CPT = \alpha k / s\Delta x^3$ (see page 91).

With $\alpha = 0.01$, $\Delta x = 0.1$, $s = 0.3$, $CPT = 100k/3$.

The computational efficiencies are tabulated below.

Scheme	OPCT	ϵ	CE
FTCS	1210	0.1634	0.184
DuFort-Frankel	930	0.0136	2.21
3L-4TH	4850	0.00416	7.21

As we would expect the 3L-4TH scheme has the highest computational efficiency even though it also has the highest operation count. Both of the

other schemes are second-order with roughly the same operation count. However the specific choice $s = 0.3$ is close to the "fourth-order accurate" value, $s = (1/12)^{0.5}$, for the DuFort-Frankel scheme. If s were close to 1/6 we would expect the FTCS scheme to demonstrate a much higher computational efficiency.

7.3 Here we undertake the construction of a two-level, five-point scheme of fourth-order accuracy. A general two-level, five-point scheme for the diffusion equation may be written as,

$$\frac{T_j^{n+1} - T_j^n}{\Delta t} - \alpha\ L_{xx}^5 T_j^n = 0\ . \tag{1}$$

It is required to find the form of $L_{xx}^5 T_j^n$ to give fourth-order accuracy .

a) Interior points:

Using a symmetric five point operator for L_{xx}^5 , we have,

$$\frac{T_j^{n+1} - T_j^n}{\Delta t} = \frac{\alpha}{\Delta x^2}(aT_{j-2} + bT_{j-1} + cT_j + bT_{j+1} + aT_{j+2})^n\ . \tag{2}$$

To find the constants a, b and c we expand every term in (2) as a Taylor series about T_j^n. Following the procedure used in Problem 3.1 we have

$$L_{xx}^5 T = \frac{2a+2b+c}{\Delta x^2} + (4a+b)T_{xx} + \frac{16a+b}{12}\Delta x^2 T_{x^4} + O(H)\ . \tag{3}$$

Clearly, $2a + 2b + c = 0$ and $4a + b = 1$, so that

$$L_{xx}^5 T = T_{xx} + \frac{16a+b}{12}\Delta x^2 T_{x^4} + O(H)\ . \tag{4}$$

Expanding the LHS of (1) also as a Taylor series about T_j^n and carrying out the usual simplification, we have

$$T_t - \alpha T_{xx} + T_{x^4}\left(\frac{\alpha^2\Delta t}{2} - \alpha\frac{16a+b}{12}\Delta x^2\right) + O(H) = 0\ .$$

Fourth-order accuracy is achieved when

$$\frac{\alpha^2\Delta t}{2} = \alpha\frac{16a+b}{12}\Delta x^2\ , \quad \text{or} \quad s = \frac{16a+b}{6}\ . \tag{5}$$

Consequently we choose $a = s/2 - 1/12$, $b = 4/3 - 2s$, $c = 3s - 5/2$.

The possible scheme for the interior points is, from (2),

$$\begin{aligned}T_j^{n+1} = 1 + s\Big\{&(0.5s - \frac{1}{12})(T_{j-2} + T_{j+2}) \\ &+ (\frac{4}{3} - 2s)(T_{j-1} + T_{j+1}) + (3s - 2.5)T_j\Big\}^n\ .\end{aligned} \tag{6}$$

Stability: The amplification factor is found to be

$$G = s\left\{2(0.5s - \frac{1}{12})\cos 2\theta + 2(\frac{4}{3} - 2s)\cos\theta + 3s - 2.5\right\} + 1 \ . \tag{7}$$

For $|G| \le 1$, $s \le 0.66$ is the stability condition.

b) Boundary points:

(i) j = 2 Let

$$L^5_{xx}T = \frac{1}{\Delta x^2}(aT_{j-1} + bT_j + cT_{j+1} + dT_{j+2} + eT_{j+3}) \ . \tag{8}$$

A Taylor series expansion shows that

$$a+b+c+d+e = 0 \ ; \ -a+c+2d+3e = 0 \ ; \quad \text{and} \quad a+c+4d+9e = 2 \ . \tag{9}$$

$$\text{Hence} \quad L^5_{xx}T = T_{xx} + (-a+c+8d+27e)\frac{\Delta x}{6}T_{x^3} + (a+c+16d+81e)\frac{\Delta x^2}{24}T_{x^4} + O(H) \ . \tag{10}$$

Substituting (10) in (1) and carrying out the truncation error analysis as before, we have

$$T_t - \alpha T_{xx} - \frac{\alpha\Delta x}{6}T_{x^3}(-a+c+8d+27e) + T_{x^4}\left\{\frac{\alpha^2\Delta t}{2} - \frac{\alpha\Delta x^2}{24}(a+c+16d+81e)\right\} + O(H) = 0 \ .$$

For a fourth-order accuracy, we require, in addition to (9),

$$-a+c+8d+27e = 0, \quad \frac{\alpha\Delta x^2}{24}(a+c+16d+81e) = \frac{\alpha^2\Delta t}{2} \quad \text{i.e.} \quad a+c+16d+81e = \frac{24s}{2} = 12s \ . \tag{11}$$

Solving (9) and (11) for a, b, c, d and e, we have

$$a = s/2 + 11/12, \ b = -2s - 5/3, \ c = 3s + 1/2, \quad d = -2s + 1/3, \ e = s/2 - 1/12 \ . \tag{12}$$

Hence the scheme at $j = 2$, comes out to be

$$T_j^{n+1} = s(\frac{s}{2} + \frac{11}{12})T^n_{j-1} + (1 - 2s^2 - \frac{5s}{3})T^n_j + s(3s + \frac{1}{2})T^n_{j+1} + s(-2s + \frac{1}{3})T^n_{j+2} + s(\frac{s}{2} - \frac{1}{12})T^n_{j+3} \ .$$

Stability: The amplification factor is

$$G = (1 - 2s^2 - \frac{5s}{3}) + s(\frac{s}{2} + \frac{11}{12})\cos\theta + s(3s + \frac{1}{2})\cos\theta + s(-2s + \frac{1}{3})\cos 2\theta + s(\frac{s}{2} - \frac{1}{12})\cos 3\theta + i\left\{[-s(\frac{s}{2} + \frac{11}{12}) + s(3s + \frac{1}{2})]\sin\theta + s(-2s + \frac{1}{3})\sin 2\theta + s(\frac{s}{2} - \frac{1}{12})\sin 3\theta\right\} \ .$$

For $|G| \le 1$, $s \le 0.36$.

(ii) j = JMAX − 1

Let the scheme be

$$\frac{T_j^{n+1} - T_j^n}{\Delta t} = \frac{\alpha}{\Delta x^2}(aT_{j-3} + bT_{j-2} + cT_{j-1} + dT_j + eT_{j+1}) \ .$$

Carrying out the analysis as before, we find the algorithm to be

$$T_j^{n+1} = s(\frac{s}{2} - \frac{1}{12})T^n_{j-3} + s(-2s + \frac{1}{3})T^n_{j-2} + s(3s + \frac{1}{2})T^n_{j-1} + (1 - 2s^2 - \frac{5s}{3})T^n_j + s(\frac{s}{2} + \frac{11}{12})T^n_{j+1} \ .$$

Alternatively we could have deduced this scheme from the boundary scheme at $j = 2$.

Stability: The amplification factor is

$$G = (1 - 2s^2 - \frac{5s}{3}) + s(\frac{s}{2} + \frac{11}{12})\cos\theta + s(3s + \frac{1}{2})\cos\theta + s(-2s + \frac{1}{3})\cos 2\theta + s(\frac{s}{2} - \frac{1}{12})\cos 3\theta + i\left\{[s(\frac{s}{2} + \frac{11}{12}) - s(3s + \frac{1}{2})]\sin\theta - s(-2s + \frac{1}{3})\sin 2\theta - s(\frac{s}{2} - \frac{1}{12})\sin 3\theta\right\}.$$

Thus $|G| \le 1$ for all $s \le 0.36$. As expected this is the same restriction as for the boundary scheme at $j = 2$. As noted in Sect. 4.3.4 the von Neumann stability method is more reliable for interior schemes than boundary schemes.

Numerical tests:

Running the computer program for various values of s and Δx, the following rms solution errors are obtained. The rate of reduction of error is indicated in brackets.

s	$\Delta x = 0.2$	$\Delta x = 0.1$	$\Delta x = 0.05$
0.1	0.1959	0.0087($\approx$ 22)	0.00022($\approx$ 40)
0.2	0.067	0.0038($\approx$ 17)	0.00022($\approx$ 17)
0.3	0.2195	0.012($\approx$ 18)	0.00061($\approx$ 20)
0.4	0.2935	0.0179($\approx$ 16)	0.00103($\approx$ 17)
0.5	0.9	0.0232($\approx$ 40)	0.00148($\approx$ 16)
0.6	1.461	0.1888($\approx$ 8)	0.00184($\approx$ 100)

We notice that a fourth-order accuracy is indeed obtained as is evident from the rates of reduction of error.

To check the stability condition, the program is run with values of s close to the stability limit. The following rms errors are obtained when $\Delta x = 0.05$, $t = 75000$ sec..

s	rms error
0.6	0.256×10^{-3}
0.65	0.256×10^{-3}
0.67	0.259×10^{-3}
0.68	0.126×10^{7}

The computed errors confirm the interior point stability limit of $s \le 0.66$, and indicate that the predicted stability limits for the boundary schemes are unnecessarily restrictive.

7.4 An inspection of Fig. 7.3 indicates that the scheme (7.13) is stable for certain values of γ. By solving (7.17) for negative values of γ, one obtains the following stability limits on s.

$\gamma =$	0	−0.1	−0.2	−0.3	−0.4	−0.5	−0.6
$s \le$	0.33	0.31	0.29	0.26	0.22	0.166	unstable

By running DIFEX with $\gamma = -0.5$ and s close to the stability limit the following solution errors are computed, at $t = 90$ sec. with $\Delta x = 0.1$.

$s =$	0.16	0.17	0.18
solution error	0.2930×10^{-4}	0.4636×10^{-1}	unstable

To check the effect of negative values of γ on the accuracy, running the program for different values of Δx and $\gamma = -0.5$, $s = 0.16$, $t = 9$ sec., we find the rms errors to be

$\Delta x =$	0.1	0.05
error	0.5743×10^{-3}	0.2845×10^{-4}

Thus we find that the error decreases at a rate of about 20, thus showing that the fourth-order accuracy of the three level scheme is preserved. One notes that the scheme (7.13) with positive or negative values of γ should give a fourth-order accuracy as is evident from (7.15) which is constructed to ensure that accuracy.

For the choice $\gamma = -0.5$, it is readily seen that the term

$$\{(1+\gamma)(T_j^{n+1} - T_j^n) - \gamma(T_j^n - T_j^{n-1})\}/\Delta t\ , \quad \text{reduces to} \quad (T_j^{n+1} - T_j^{n-1})/2\Delta t\ ,$$

as for the DuFort-Frankel scheme. Thus scheme (7.13) now looks like a DuFort-Frankel scheme, ie.

$$\frac{T_j^{n+1} - T_j^{n-1}}{2\Delta t} - \alpha\left[(1-\beta)L_{xx}T_j^n + \beta\, L_{xx}T_j^{n-1}\right] = 0\ . \tag{1}$$

In (1) the spatial discretisation does not contain T^{n+1} terms unlike the DuFort-Frankel scheme (see 7.9). But by setting $\beta = 0$, one gets the scheme (7.8) which is unconditionally unstable.

7.5 The condition for fourth-order accuracy, (7.30), in the scheme (7.26) will be derived. The scheme (7.26) is

$$\begin{aligned}(1+\gamma)M_x\left(\frac{\Delta T_j^{n+1}}{\Delta t}\right) - \gamma M_x\left(\frac{\Delta T_j^n}{\Delta t}\right)&\\ -\alpha\left[\beta\, L_{xx}T_j^{n+1} + (1-\beta)L_{xx}T_j^n\right] = 0\ ,&\end{aligned} \tag{1}$$

with $\Delta T_j^{n+1} = T_j^{n+1} - T_j^n$, $\Delta T_j^n = T_j^n - T_j^{n-1}$ and $L_{xx}T_j = (T_{j-1} - 2T_j + T_{j+1})/\Delta x^2$.

Expanding the terms as a Taylor series about T_j^n we have

$$\begin{aligned}\frac{(1+\gamma)M_x\Delta T_j^{n+1}}{\Delta t} - \frac{\gamma M_x \Delta T_j^n}{\Delta t} &= M_xT_t + M_x(1+2\gamma)\frac{\Delta t}{2}T_{tt}\\ &+ M_x\frac{\Delta t^2}{6}T_{t^3} + M_x(1+2\gamma)\frac{\Delta t^3}{24}T_{t^4} + O(H)\ .\end{aligned} \tag{2}$$

Substituting for M_x and expanding each group of terms about T_j^n and simplifying, (2) reduces to

$$T_t + (2\gamma+1)\frac{\Delta t}{2}T_{tt} + \frac{\Delta t^2}{6}T_{t3} + \frac{\Delta t^3}{24}T_{t4} + \delta\Delta x^2 T_{txx} + O(H)\,. \qquad (3)$$

Following a similar procedure we find

$$\alpha[\beta L_{xx}T_j^{n+1} + (1-\beta)L_{xx}T_j^n] = \alpha[T_{xx} + \frac{\Delta x^2}{12}T_{x4} + \beta\Delta t T_{xxt}] + O(H). \quad (4)$$

Substituting (3) and (4) in (1), we have

$$T_t - \alpha T_{xx} + \alpha\Delta x^2\, T_{x4}\, s\left(\frac{2\gamma+1}{2} + \frac{\delta}{s} - \frac{1}{12s} - \beta\right) + O(H) = 0\,.$$

For fourth-order accuracy,

$$\beta = \frac{2\gamma+1}{2} + \frac{\delta}{s} - \frac{1}{12s}\,,$$

as required.

7.6 Stability of the scheme (7.31) is analysed as follows.

The given scheme is

$$\begin{aligned}A_jT_{j+1}^{n+1} + B_jT_j^{n+1} + C_jT_{j-1}^{n+1} &= \\ (1+2\gamma)M_xT_j^n - \gamma M_xT_j^{n-1} &+ (1-\beta)sL'_{xx}T_j^n.\end{aligned} \qquad (1)$$

Writing out (1) in an extended form and substituting for A, B, C and L'_{xx} from (7.31), the 'error equation' is

$$\begin{aligned}&\{(1+\gamma)\delta - s\beta\}\{\xi_{j-1}^{n+1} + \xi_{j+1}^{n+1}\} + \{(1+\gamma)(1-2\delta) + 2s\beta\}\xi_j^{n+1} \\ &= (1+2\gamma)\big(\delta\xi_{j-1} + (1-2\delta)\xi_j + \delta\xi_{j+1}\big)^n \\ &- \gamma\big(\delta\xi_{j-1} + (1-2\delta)\xi_j + \delta\xi_{j+1}\big)^{n-1} \\ &+ (1-\beta)s\big(\xi_{j-1} - 2\xi_j + \xi_{j+1}\big)^n.\end{aligned} \qquad (2)$$

Upon substituting $\xi_j^n = G^n e^{i\theta j}$, the amplification factor is given by

$$\begin{aligned}&G^2\{((1+\gamma)\delta - s\beta)(2\cos\theta) + (1+\gamma)(1-2\delta) + 2s\beta\} \\ &- G\{(1+2\gamma)(2\delta\cos\theta + 1 - 2\delta) + (1-\beta)(2\cos\theta - 2\} \\ &+ \gamma\{2\delta\cos\theta + 1 - 2\delta\} = 0\,.\end{aligned} \qquad (3)$$

Also, from (7.30),

$$\beta = 0.5 + \gamma + \frac{\delta - \frac{1}{12}}{s}\,. \qquad (4)$$

A program was written to calculate G for given values of δ, γ and s. The stability limit obtained for various δ and negative values of γ values are given in the following table.

γ	-0.5	-0.4	-0.3	-0.2	-0.1	0
$\delta = 0$	unstable	unstable	$s \le .055$	$s \le .33$	$s \le 1.16$	$s \le 9.9$
$\delta = 1/12$	unstable	$s \le .083$	$s \le .22$	$s \le .5$	$s \le 1.3$	$s \le 10$
$\delta = 1/6$	$s \le .15$	$s \le .23$	$s \le .37$	$s \le .65$	$s \le 1.4$	$s \le 9.5$

7.7 DIFIM is run with $\delta = 1/6$, $\gamma = 0$, $\beta = 1$ and $s = 1/6$ as suggested. Note that this is the case $ME = 4$ for which DIFIM employs the FEM-4TH order scheme.

The modified program was run for the conditions: $\Delta x = 0.1$, $t = 12$ sec., $TST = 4.5$ sec., and the computed solution error was 0.3068×10^{-3}. For comparison the FTCS gives an error of 0.6872×10^{-2}. This is additional evidence of the fact that FEM with mass operators gives substantially more accurate results than a finite difference scheme.

Other parametric combinations that make (7.31) explicit:

All $A_j = C_j = 0$ i.e. $\beta = (1+\gamma)\delta/s$.

Substituting this into (7.30) gives

$\gamma = (0.5s - 1/12)/(\delta - s)$.

7.8 Running the program DIFIM for the cases suggested, i.e. $ME = 4, 5$ and $\gamma = 0.5$ and $s = 1.0$, the following results are obtained. In the table r denotes the rate of convergence.

$ME = 4$, **FEM-4th order**

Δx	solution error	r
0.2	0.8524	
0.1	0.0554	≈ 3.9
0.05	0.0035	≈ 4
0.025	0.0002	≈ 4

$ME = 5$, **Composite scheme**

Δx	solution error	r
0.2	1.085	
0.1	0.0636	≈ 4.1
0.05	0.004	≈ 4
0.025	0.0002	≈ 4

Comparing these results with those in Table 7.6, we find that for a given Δx the error increases with γ. The convergence rate, however, is about the same for all values of γ. The result is in accordance with the observation made in the text (page 235).

7.9 Operation counts will be obtained for the implicit schemes. For the program DIFIM, the approximate operation counts are

$$(JMAX-2)(3FL+2FX)+N_t(JMAX-2)\{1FX+3(7FX+12FL+1R)\}$$
$$+N_t(5FL)+N_t(JMAX-2)(3FL+3FX+1R)$$
$$+(JMAX-3)N_t(6FL+9FX+3R) \qquad \text{(in subroutine BANSOL)}$$
$$+(JMAX-3)(8FX+3FL+1R) \qquad \text{(in BANFAC)}\ .$$

which simplifies to

$$\begin{aligned}&(6.6JMAX-2.7)+\\&N_t\{(JMAX-2)(40.75)+5+(JMAX-3)(6.75)\}FL_{eq}\ .\end{aligned} \tag{1}$$

Running the program DIFIM for $ME=1$ and 4, and with $s=1.0$, TST=4.5 sec. and t=12 sec., we find that each of the cases takes 8 time steps, when $\Delta x=0.1$ i.e. $JMAX=11$.

Thus the operation counts are, from (1),

$$(6.6\times 11-2.7)+8\{9\times 40.75+s+8\times 6.75\}=3470FL_{eq}\ .$$

Computational efficiency: Running the programs DIFEX and DIFIM for the suggested cases, the following solution errors are obtained. Computational efficiencies are calculated following the procedure outlined for Problem 4.14.

Scheme	Solution error	Comp. efficiency
FTCS(s=0.5)	0.3264	0.1532
DuFort-Frankel(s=0.3)	0.01076	2.788
DIFIM (ME=1, FDM-2ND, s=1.0)	0.1466	0.682
DIFIM (ME=4, FEM-4TH, s=1.0)	0.0152	6.57

The FEM-4TH order scheme is seen to be the most efficient. This is a consequence of the fact that this scheme gives a low error at a fairly high s value of 1.0.

7.10 Modifications of the program DIFEX are made to implement Neumann boundary conditions. The test case chosen is the one described in Section 7.3.2 of the text. The following changes are made to DIFEX.

1. The region of interest now is $0.1 \le x \le 1.0$.

2. The subroutine EXACT is modified since now the exact solution is simply given by (7.40).

3. The Neumann conditions are forced at $x=0.1$. An option is built into the program to choose the first-order implementation (7.32) or the second-order one (7.34).

a) First-order Neumann conditions: Running the program with $s=0.3$ for different values of Δx up to a time level of $t\approx 9$ sec. gives the following solution errors.

Scheme \ $\Delta x =$	0.225	0.1125	0.05625
FTCS+(7.32)	0.2028	0.0812	0.0357
ME = 2	0.2205	0.08312	0.0358
ME =3	0.1952	0.0721	0.0309

For the FTCS scheme (i.e. $ME=1$) the error distribution, with $JMAX=5$, is as follows

$x=$	0.1	0.325	0.55	0.775	1.0
Error	0.3937	0.2060	0.08657	0.027	0.0

For $ME=1$, i.e. for the FTCS scheme, it is observed that the solution error and the error distribution closely follows the ones given in Tables 7.8 and 7.9.

b) Second-order Neumann conditions: For $s=0.3$, $t\approx 9.0$ sec.

Scheme \ $\Delta x =$	0.225	0.1125	0.05625
FTCS	0.00175	0.000420	0.000107
DuFort-Frankel	0.00152	0.000445	0.000118
3L-4TH order, γ =0	0.00176	0.000512	0.000136

The error distribution, with $JMAX=5$, is as indicated.

Scheme \ x=	0.1	0.325	0.55	0.775	1.0
FTCS	-0.00033	-0.00221	-0.00271	-0.00173	0.0
DuFort-Frankel	-0.00309	-0.00137	-0.000355	-0.0000667	0.0
3L-4TH order ,$\gamma=0$	-0.00344	-0.00177	-0.000755	-0.000268	0.0

The rms solution error and the error distribution are seen to agree closely with the ones given in Tables 7.8 and 7.9. For the 3L-4TH scheme the errors computed are smaller than those shown in Table 7.9.

Modifications to DIFEX for Problem 7.10

1. The variable *NORDER* is read in along with the others.
2. The following statements replace lines 12 to 16.

```
      PI = 3.1415927
      PI2=2.*PI
      EXTERM=-ALPH*PI*PI/4.
      SINP=SIN(0.05*PI)
      XINT=0.1
      XEND=1.0
      JMAP = JMAX - 1
      AJM = JMAP
      DELX=(XEND-XINT)/(JMAX-1)
      DELT = DELX*DELX*S/ALPH
```

3. Lines 53 and 54 are replaced by

```
       IF(I .EQ. 1)TOL(J) = TE(J)
      IF(I .EQ. 2)TN(J) = TE(J)
```

4. Lines 60 to 63 are deleted.
5. Lines 95 to 102 are replaced by

```
C
   17 CONTINUE
        IF(NORDER.EQ.2) THEN
        CN= (2. - PI2*SINP*EXP(EXTERM*T))
        DUM(1)=-2.*S*DELX*CN+(1.-2.*S)*TN(1)+2.*S*TN(2)
        ENDIF
C
      DO 18 J = 2,JMAP
      IF(ME .GT. 1)TOL(J) = TN(J)
   18 TN(J) = DUM(J)
C
        IF(NORDER.EQ.2) THEN
        IF(ME.GT.1) TOL(1)=TN(1)
        TN(1)=DUM(1)
        ENDIF
C
C     SET BOUNDARY CONDITIONS
C
      T = T + DELT
        IF(NORDER .EQ. 1) THEN
        TN(1)=TN(2)-DELX* (2. - PI2*SINP*EXP(EXTERM*T))
        TOL(1) = TN(1)
        ENDIF
C
      TN(JMAX) = 2.
      TD(1) = TN(1)
      TD(JMAX) = TN(JMAX)
```

6. Subroutine EXTRA is modified as

```
      Subroutine EXTRA(JMAX,MAXEX,DELX,PI,ALPH,T,TE)
C
C     EXACT SOLUTION OF THE TRANSIENT HEAT CONDUCTION PROBLEM
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION X(41),TE(41)
C
      XINT=0.1
      XEND=1.0
      EXTERM=EXP(-ALPH*PI*PI*T/4.)
      HAFPI=PI/2.
      DO 2 J = 1,JMAX
      X(J) =XINT+ (XEND-XINT)*(J-1)/(JMAX-1)
      TE(J) = 2.*X(J)+4.*COS(HAFPI*X(J))*EXTERM
    2 CONTINUE
      RETURN
      END
```

7.11 DIFIM is modified to implement the Neumann boundary condition

$$\frac{\partial T}{\partial x} = c \quad at\ x = 0.1\ . \tag{1}$$

a) **First-order Neumann conditions:**

The algebraic equation used is

$$T_1^{n+1} = T_2^{n+1} - C^{n+1}\Delta x\ , \tag{2}$$

which is a discretised version of (1). See also (7.33) of the text.

In program DIFIM, the coefficients $B(1)$, $C(1)$ and $D(1)$ are modified accordingly.

The computed solution errors are tabulated below for $s = 1.0$, $t \approx 15$ sec.

Scheme \ $\Delta x =$	0.225	0.1125	0.05625
FDM-2ND , ME=1	0.1655	0.0758	0.0344

The corresponding error distribution for $\Delta x = 0.225$ is

$x =$	0.1	0.325	0.55	0.775	1.0
Error	0.3264	0.1622	0.01627	0.00872	0.0

b) Second-order Neumann conditions:

To implement the second-order Neumann conditions at $x = 0.1$, (7.36) is employed. The modifications to program DIFIM are shown in the following listing. The computed solution errors for $s = 1.0$, $t = 15$ sec. are tabulated below.

Scheme \ Δx =	0.225	0.1125	0.05625
FDM-2ND , ME=1	0.0318	0.0027	0.000543
FDM-4TH , γ =0	0.0108	0.00177	0.000293
FDM-4TH , γ =1	0.0097	0.00161	0.000280

The error distribution for $\Delta x = 0.225$ is

Scheme \ x=	0.1	0.325	0.55	0.775	1.0
FDM-2ND	-0.02585	-0.01469	-0.00756	-0.003266	0.0
FDM-4TH,γ =0	-0.02185	-0.00984	-0.0032	-0.000808	0.0
FDM-4TH,γ =1	-0.0202	-0.00722	-0.0016	-0.0	0.0

The errors, both 'rms' and 'local', are smaller on the fine grid as expected.

Modifications to DIFIM for Problem 7.11

1. The variable *NORDER* is read in along with the others.
2. The following statements replace lines 15 to 19.

```
      PI = 3.1415927
      PI2=2.*PI
      EXTERM=-ALPH*PI*PI/4.
      SINP=SIN(0.05*PI)
      XINT=0.1
      XEND=1.0
      JMAP = JMAX - 1
      AJM = JMAP
      DELX=(XEND-XINT)/(JMAX-1)
      DELT = DELX*DELX*S/ALPH
```

3. Lines 65 and 66 are replaced by

```
      IF(I .EQ. 1)TOL(J) = TE(J)
      IF(I .EQ. 2)TN(J) = TE(J)
```

4. Lines 76 to 77 are replaced by,

```
      TD(1) = TN(1)
      TD(JMAX) = TN(JMAX)
```

5. Lines 87 to 102 are replaced by,

```
C
      IF(NORDER .EQ. 2) THEN
      A(1,1) = 0.
      A(2,1) = 0.
      A(3,1) = 1.+2.*S
      A(4,1) = -2.*S
      A(5,1) = 0.
      ELSE
C
      A(1,1) = 0.
      A(2,1) = 0.
      A(3,1) = 1.
      A(4,1) = -1.
      A(5,1) = 0.
      ENDIF
C
      DO 14 J = 2,JMAP
      JM = J
      IF(N .GT. 1)GOTO 12
      A(1,JM) = 0.
      A(2,JM) = AD
      A(3,JM) = BD
      A(4,JM) = CD
      A(5,JM) = 0.
   12 D(JM) = 0.
C
      DO 13 K = 1,3
      KJ = J - 2 + K
      D(JM) = D(JM) + EMX(K)*((1. + 2.*GAM)*TN(KJ)
     1  - GAM*TOL(KJ))
      D(JM) = D(JM) + S*ELX(K)*(1.-BET)*TN(KJ)
   13 CONTINUE
   14 CONTINUE
C
      T = T + DELT
      CN=2. - PI2*SINP*EXP(EXTERM*T)
      IF(NORDER .EQ. 2) THEN
      D(1)=TN(1)-2.*S*DELX*CN
      ELSE
C
      D(1)=-CN*DELX
      ENDIF
      D(JMAP) = D(JMAP) - A(4,JMAP)*TN(JMAX)
C
C     SOLVE BANDED SYSTEM OF EQUATIONS
```

6. Subroutine EXTRA is modified as for Problem 7.10.

7.12 Here we consider the effect of different initial conditions on the solution obtained from Dufort-Frankel scheme. The DuFort-Frankel scheme needs two levels of starting data: namely at t_o and $t_o - \Delta t$. The resulting solution is to be compared with that obtained by introducing the FTCS scheme at $t_o - \Delta t$. The program DIFEX with $ME = 2$ is used to get the DuFort-Frankel solution with both the levels of starting data being the exact solutions.

a) The solution errors at $t = 9$ sec. are given below for various values of s and Δx, with the two levels of starting data given by the exact solution.

$s \quad \backslash\Delta x =$	0.2	0.1	0.05	0.025
0.3	0.02437	0.01362	0.003955	0.001029
0.5	1.732	0.4099	0.1032	0.02602

b) The solution errors for the case when the starting data at $t = t_o - \Delta t$ is exact and that at t_o is computed by the FTCS scheme.

$s \quad \backslash\Delta x =$	0.2	0.1	0.05	0.025
0.3	0.1148	0.01845	0.004263	0.001048
0.5	1.732	0.4099	0.1032	0.02602

As expected, the solution for Case a) is more accurate than that for Case b). The inferior accuracy for the Case b) is accounted for by the fact that the starting data already has the error introduced by the FTCS scheme. However, as the grid is refined, this error in the starting data is reduced and the two cases tend to be equally accurate.

The solutions at $s = 0.5$ are identical for the two cases. This is due to the fact that the DuFort-Frankel scheme reduces to the FTCS scheme when $s = 0.5$ (see p. 220).

7.13 The program DIFEX is readily modified to apply the four-stage Runge Kutta method (details given in Sect. 7.4). A listing of the program is given after Problem 7.14.

Running the program with $s = 0.41$, the computed solution errors are as follows $s = 0.41$, $t_{st} = 2$ sec., $t = 9$ sec.

$\Delta x=$	0.2	0.1	0.05
Error	0.8	0.2037	0.05151

It is noted that the error decreases by a factor of 4 as the grid size is halved indicating that the method is of second-order accuracy. This implies that the overall accuracy is dominated by the discretisation of the spatial term.

Comparing with the results in Table 7.3, it is found that the four-stage Runge-Kutta method is more accurate than the FTCS scheme and is of comparable accuracy to the DuFort-Frankel scheme. The accuracy of the Runge-Kutta method is inferior to that of the three-level fourth-order scheme.

7.14 i) Truncation error analysis for the four-stage Runge-Kutta method: Denote the scheme by

$$T_j^{n+1} = T_j^n + \frac{\Delta t}{6}(f^n + 2f^* + 2f^{**} + f^{***}) , \tag{1}$$

where

$$T^* = T_j^n + \alpha_1 \; \Delta t \; f^n ,$$
$$T^{**} = T_j^n + \alpha_2 \; \Delta t \; f^* ,$$
$$T^{***} = T_j^n + \alpha_3 \; \Delta t \; f^{**} ,$$

and $\alpha_1 = \alpha_2 = 0.5$, $\alpha_3 = 1.0$.

$$f^n = \frac{\bar{\alpha}}{\Delta x^2}(T_{j+1}^n - 2T_j^n + T_{j-1}^n) .$$

Applying a Taylor series expansion the following expressions are obtained.

$$f^n(T) = \bar{\alpha}(T_{xx} + \frac{\Delta x^2}{12}T_{x^4}) + O(H) . \tag{2}$$

$$T^* = T_j^n + \alpha_1\bar{\alpha}\Delta t \; T_{xx} + \alpha_1\bar{\alpha}\frac{\Delta t \; \Delta x^2}{12}T_{x^4} + O(H) .$$

$$f^* = \bar{\alpha}(T_{xx} + \alpha_1\bar{\alpha}\Delta t \; T_{x^4} + \frac{\Delta x^2}{12}T_{x^4}) + O(H) . \tag{3}$$

$$T^{**} = T_j^n + \bar{\alpha}\alpha_2 \; \Delta t \; T_{xx} + \alpha_1\alpha_2\bar{\alpha}^2\Delta t^2 \; T_{x^4} + \frac{\bar{\alpha}\alpha_2 \; \Delta t \; \Delta x^2}{12}T_{x^4} + O(H) .$$

$$f^{**} = \bar{\alpha}(T_{xx} + \bar{\alpha}\alpha_2 \; \Delta t \; T_{x^4} + \frac{\Delta x^2}{12}T_{x^4}) + O(H) . \tag{4}$$

$$T^{***} = T_j^n + \alpha_3\bar{\alpha}\Delta t \; T_{xx} + \bar{\alpha}^2\alpha_2\alpha_3 \; \Delta t^2 \; T_{x^4} + \frac{\alpha_3\bar{\alpha}}{12}\Delta t \; \Delta x^2 \; T_{x^4} + O(H) .$$

$$f^{***} = \bar{\alpha}[T_{xx} + \alpha_3\bar{\alpha}\Delta t \; T_{x^4} + \frac{\Delta x^2}{12}T_{x^4}] + O(H) . \tag{5}$$

Substituting (2), (3), (4) and (5) in (1) and expanding T_j^{n+1} as a Taylor series, we get

$$T_t - \bar{\alpha} T_{xx} = \bar{\alpha}\left(\frac{\Delta x^2}{12} + \frac{2}{3} s\, \Delta x^2 - \frac{s}{2}\Delta x^2\right) + O(H)\,.$$

where the relations $T_{tt} = \alpha^2 T_{x^4}$ and $s = \bar{\alpha}\Delta t/\Delta x^2$ have been used. The leading term in the truncation error is thus found to be

$$\bar{\alpha} T_{x^4}\left(\frac{1}{12} + \frac{s}{6}\right)\Delta x^2.$$

Thus the method is of second-order accuracy as already observed in Problem 7.13.

ii) von Neumann stability analysis of the Runge-Kutta method:

As described in Sect. (7.4) the Runge-Kutta methods have the stability interval given by

$-2.78 \le \lambda_k \Delta t \le 0$.

The eigenvalues λ_k are given by (7.58)

$$\lambda_k = \alpha(-2 + 2\cos\frac{k\pi}{M+1})/\Delta x^2\,.$$

Since we are considering a scalar equation $M = 1$ and $\max|\lambda_k| = 4\alpha/\Delta x^2$.

Hence for stability we have $4\,\alpha\Delta t/\Delta x^2 \le 2.78$ or $s \le 2.78/4 \le 0.7$.

To check the stability analysis the program RUNKUT, listing given later, was run with s values closer to the above determined stability limit and the computed solution errors for $\Delta x = 0.05$, $t_{st} = 2$ sec., $t = 90$ sec. are as follows.

s	Error
0.41	0.791×10^{-12}
0.7	0.219×10^{-3}
0.71	10^5

Thus the predicted stability limit is obtained in practice.

iii) Operation count and computational efficiency for the four-stage Runge-Kutta method:

The operation count for the Runge-Kutta method is (see listing below)

$$\Big\{2FL + 8\,FX + 2R + 4(JMAX-2)N_t[8FL + 7FX + 3R + FL + FX]$$
$$+ N_t(JMAX-2)(2FL + 1FX + FL + 2FX + R)$$
$$+ N_t(FL + R)\Big\}FL_{eq},$$

which simplifies to $\{2.6 + N_t(JMAX-2)(42) + 1.1\,N_t\}FL_{eq}$.

Running the program with $s = 0.41$, $\Delta x = 0.1$, one finds that it requires 18 time steps to reach a time level of 9 sec. starting from a time level of 2 sec.

The operation counts are 6826 FL_{eq}.

The solution error is 0.2037 so that the computational efficiency is 0.201.

From Table (7.3) we find that the computational efficiency, for s=0.41, of the FTCS, DuFort-Frankel and three level fourth-order schemes to be 0.1356, 0.1966 and 1.79. Thus we observe that the Runge-Kutta scheme demonstrates comparable efficiency to that of the other second-order schemes. But as noted in Problem 7.13 the discretisation of the spatial term is the dominant factor.
Program RUNKUT is obtained by replacing lines 1 to 112 of DIFEX with the following lines.

Modifications to DIFEX for Problems 7.13 and 7.14

```
C     PR7P13, PR7P14.
C     RUNKUT SOLVES THE DIFFUSION
C     (1D TRANSIENT HEAT CONDUCTION)
C     EQUATION USING RUNGE KUTTA METHOD.
C
C     WRITTEN TO RUN FOR VARIOUS VALUES OF S.
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION TN(41),DUM(41),TD(41),TE(41),TOL(41)
     1 ,COEFF(4),BETA(4),RHS(41),TF(41)
C
      OPEN(1,FILE='RUNKUT.DAT')
      OPEN(6,FILE='RUNKUT.OUT')
      READ(1,*)IPR,JMAX,MAXEX,NMAX,ALPH,S,
     1 TMAX,TST,NSTAGE
C
  555 WRITE(*,*)'GIVE S>'
      READ(*,*)S
      IF(S.EQ.999.) GO TO 999
C
      PI = 3.1415927
      JMAP = JMAX - 1
      AJM = JMAP
      DELX = 1./AJM
      DELT = DELX*DELX*S/ALPH
C
      WRITE(6,44)
   44 FORMAT(' RUNGE - KUTTA SCHEME',//)
      WRITE(6,5)JMAX,MAXEX,NMAX,TMAX,TST
    5 FORMAT(' JMAX=',I5,'  MAXEX=',I5,'  NMAX=',I5,
     1'  TMAX=',F5.2,'  TST=',F5.2)
      WRITE(6,6)S,ALPH,DELT,DELX
    6 FORMAT(' S=',F5.3,'  ALPH = ',E10.3,'  DELT = ',E10.3,
     1'  DELX = ',E10.3,//)
C
C     COEFFICIENTS FOR THE RUNGE KUTTA METHOD
C     ( SEE EQN. 7.53)
C
      COEFF(1)=0.5
      COEFF(2)=0.5
      COEFF(3)=1.0
      COEFF(4)=1.0
C
      BETA(1)=1./6.
      BETA(2)=1./3.
      BETA(3)=1./3.
      BETA(4)=1./6.
C
C     OBTAIN INITIAL CONDITIONS FROM EXACT SOLUTION
C
      T = TST
      CALL EXTRA(JMAX,MAXEX,DELX,PI,ALPH,T,TE)
      DO 8 J = 2,JMAP
      TOL(J) = TE(J)/100.
      TN(J) = TE(J)/100.
      TF(J) = TE(J)/100.
    8 CONTINUE
C
C     SET BOUNDARY CONDITIONS
C
      TOL(1) = 1.
      TOL(JMAX) = 1.
      TN(1) = 1.
      TN(JMAX) = 1.
      TD(1) = 100.*TN(1)
      TD(JMAX) = 100.*TN(JMAX)
      N = 0
```

```
C
C     EACH TIME STEP STARTS AT STATEMENT 10
C
   10 N = N + 1
C
C     RUNGE- KUTTA METHOD.
C
      DO 66 NST=1,NSTAGE
      DO 67 J=2,JMAP
      RHS(J)=  S*(-2.*TN(J)+TN(J-1)+TN(J+1))
C
      DUM(J) =TOL(J)+COEFF(NST)*RHS(J)
      TF(J)=TF(J)+BETA(NST)*RHS(J)
   67 CONTINUE
      DO 68 J=2,JMAP
   68 TN(J)=DUM(J)
   66 CONTINUE
      DO 69 J=2,JMAP
      TN(J)=TF(J)
   69 TOL(J)=TN(J)
C
      DO 19 J = 2,JMAP
   19 TD(J) = 100.*TN(J)
      T = T + DELT
      IF(IPR .EQ. 1)WRITE(6,20)T,(TD(J),J=1,JMAX)
   20 FORMAT(' T= ',F5.2,' TD=',11F6.2)
C
C     IF MAXIMUM TIME OR MAXIMUM NUMBER OF TIME-STEPS
C     EXCEEDED EXIT FROM LOOP
C
      IF(N .GE. NMAX)GOTO 21
      IF(T .LT. TMAX)GOTO 10
```

CTFD Solutions Manual: Chapter 8

Multidimensional Diffusion Equation

8.1 The von Neumann stability analysis of (8.7) is carried out as follows.

Since $\alpha = \alpha_x = \alpha_y$ and $\Delta x = \Delta y$, we have $s = s_x = s_y$.

Following the same procedure as for the von Neumann stability analysis carried out in the previous chapters, we see that the amplification factor is given by

$$(G-1) - 2s(\cos\theta_x - 1) - 2s(\cos\theta_y - 1) - 4s^2(\cos\theta_x - 1)(\cos\theta_y - 1) = 0 \ . \quad (1)$$

Since $\theta = \theta_x = \theta_y$,

$$G = 1 + 4s(\cos\theta - 1) + 4s^2(\cos^2\theta - 2\cos\theta + 1) \ . \quad (2)$$

$|G| \leq 1$ for $0 \leq s \leq 0.5$ as required.

8.2 Truncation error analysis for scheme 8.7 is carried out in the following way. The scheme 8.7 is given by

$$T^{n+1}_{j,k} - T^n_{j,k} - \alpha\Delta t L_{xx} T^n_{j,k} - \alpha\Delta t L_{yy} T^n_{j,k} - \alpha^2\Delta t^2 L_{xx}L_{yy} T^n_{j,k} = 0 \ . \quad (1)$$

Using centred differences and expanding the terms as a Taylor series about $T^n_{j,k}$, we get after simplification,

$$\begin{aligned} & T_t + \frac{\Delta t}{2}T_{tt} + \frac{\Delta t^2}{6}T_{ttt} + ... \\ & - \alpha(T_{xx} + \frac{\Delta x^2}{12}T_{x^4} + \frac{\Delta x^4}{360}T_{x^6} + ...) \\ & - \alpha(T_{yy} + \frac{\Delta y^2}{12}T_{y^4} + \frac{\Delta y^4}{360}T_{y^6} + ...) \\ & - \alpha^2(\Delta t T_{x^2y^2} + \Delta t\frac{\Delta x^2}{12}T_{x^4y^2} + \Delta t\frac{\Delta y^2}{12}T_{x^2y^4} + ...) \\ & = 0 \ . \end{aligned} \quad (2)$$

Further, $\dfrac{\partial T}{\partial t} = \alpha(T_{xx} + T_{yy})$, $\quad (3)$

so that

$$T_t - \alpha T_{xx} - \alpha T_{yy} + \left(\alpha^2\frac{\Delta t}{2} - \alpha\frac{\Delta x^2}{12}\right)T_{x^4} + \left(\alpha^2\frac{\Delta t}{2} - \alpha\frac{\Delta y^2}{12}\right)T_{y^4} + \\ + O(H) = 0 \ . \quad (4)$$

The leading term in the truncation error is given by

$$\left(\frac{s\alpha}{2} - \frac{\alpha}{12}\right)\Delta x^2 T_{x^4} + \left(\frac{s\alpha}{2} - \frac{\alpha}{12}\right)\Delta y^2 T_{y^4}.$$

If $s = 1/6$, the above term vanishes. Thus, the scheme should be capable of achieving fourth-order accuracy.

8.3 Program TWDIF is easily modified to apply the Mitchell-Griffith scheme given by (8.8) and (8.9). The listing of the program is given later. Running the program for the values of $s(= s_x = s_y)$ close to the theoretical stability limit of $s \leq 0.5$, the following solution errors are obtained for $t_{max} = 0.24 \times 10^5$ secs.

$\Delta x \backslash s =$	0.5	0.55	0.6	0.7
0.2	0.063	0.0724	0.0868	1.088
0.1	0.0135	0.6828	0.33×10^5	

These results clearly show that the scheme is unstable for $s \geq 0.5$.

To validate the findings in Problem 8.2, the program is run for s = 0.5 and s= 1/6 and for different mesh sizes. The computed solution errors are as follows, for $t_{max} = 0.24 \times 10^5$ secs.

$\Delta x, \Delta y =$	0.2	0.1	0.05
s=0.5	0.063	0.01349	0.003167
s=1/6	0.66×10^{-4}	0.366×10^{-5}	0.21×10^{-6}

A rate of convergence of about 4 is evident when s = 1/6 thus validating the findings of Problem 8.2.

Modifications to TWDIF for Problem 8.3

The changes made to TWDIF are as follows. Now array B is not needed and lines 36 to 42, 46 to 49, 75 to 78, 86, 87 are deleted. ME, GAM and BET are not required and not read in (line 12). Lines 93 to 161 are replaced by the following:

```
C
      DO 21 K = 2,NYN
      DO 21 J = 2,NX
```

```
   21 R(K,J)=T(K,J)+SY*(T(K-1,J) -2.*T(K,J) + T(K+1,J))
      DO 22 K = 2,NYN
      DO 22 J = 2,NX
   22 T(K,J)=R(K,J)
C
C     SOLVE FOR T(N+1), EQN 8.9.
C
      DO 23 K = 2,NYN
      DO 23 J = 2,NXN
   23 R(K,J)=T(K,J)+SX*(T(K,J-1) -2.*T(K,J) + T(K,J+1))
      DO 24 K = 2,NYN
      DO 24 J = 2,NXN
   24 T(K,J)=R(K,J)
C
```

8.4 a) The three-dimensional equivalent of the ADI scheme (8.14) and (8.15) is given by

$$\frac{T^*_{j,k,l}-T^n_{j,k,l}}{\Delta t/3} - \alpha_x\, L_{xx}\, T^*_{j,k,l} - \alpha_y L_{yy} T^n_{j,k,l} - \alpha_z L_{zz} T^n_{j,k,l} = 0 \;, \tag{1}$$

$$\frac{T^{**}_{j,k,l}-T^*_{j,k,l}}{\Delta t/3} - \alpha_x\, L_{xx}\, T^*_{j,k,l} - \alpha_y L_{yy} T^{**}_{j,k,l} - \alpha_z L_{zz} T^*_{j,k,l} = 0 \;, \tag{2}$$

$$\frac{T^{n+1}_{j,k,l}-T^{**}_{j,k,l}}{\Delta t/3} - \alpha_x\, L_{xx}\, T^{**}_{j,k,l} - \alpha_y L_{yy} T^{**}_{j,k,l} - \alpha_z L_{zz} T^{n+1}_{j,k,l} = 0 \;. \tag{3}$$

These equations may also be written in an algorithmic form as in (8.14) and (8.15).

Carrying out a von Neumann stability analysis, we can arrive at an amplification factor for each step. As in (8.16) the amplification factor for the entire scheme is given by the product of those for the individual steps.

Thus we have from (1) (2) and (3),

$$\begin{aligned} G = &\left(\frac{3/4 - s_y \sin^2\theta_y/2 - \; s_z\sin^2\theta_z/2}{3/4 + s_x\sin^2\theta_x/2}\right) \\ \times &\left(\frac{3/4 - s_x \sin^2\theta_x/2 - s_z\sin^2\theta_z/2}{3/4 + s_y\sin^2\theta_y/2}\right) \\ \times &\left(\frac{3/4 - s_x \sin^2\theta_x/2 - s_y\sin^2\theta_y/2}{3/4 + s_z\sin^2\theta_z/2}\right) . \end{aligned} \tag{4}$$

To check stability, consider the worst case, i.e. let $\sin^2\theta_x/2 = \sin^2\theta_y/2 = \sin^2\theta_z/2 = 1$ and $s_x = s_y = s_z = s$.

Then we have,

$$G = \left(\frac{1-8s/3}{1+4s/3}\right)^3 . \tag{5}$$

It is easily seen that $|G| \le 1$ for $s \le 1.5$.

Hence the ADI scheme, for this application, is stable for s_x, s_y, $s_z \le 1.5$.

b) The three-dimensional equivalent of the approximate factorisation scheme (8.23) and (8.24) is

$$\begin{aligned} (1-\beta\Delta t\alpha_x L_{xx})(1-\beta\Delta t\alpha_y L_{yy})(1-\beta\Delta t\alpha_z L_{zz})\Delta T^{n+1}_{j,k,l} = \\ \Delta t(\alpha_x L_{xx} + \alpha_y L_{yy} + \alpha_z L_{zz})T^n_{j,k,l} \;. \end{aligned} \tag{6}$$

Carrying out the von Neumann analysis we have

$$\begin{aligned} &\left(1+4\beta s_x\sin^2\frac{\theta_x}{2}\right)\left(1+4\beta s_y\sin^2\frac{\theta_y}{2}\right)\left(1+4\beta s_z\sin^2\frac{\theta_z}{2}\right)(G-1) \\ &= -\left(4s_x\sin^2\frac{\theta_x}{2} + 4s_y\sin^2\frac{\theta_y}{2} + 4s_z\sin^2\frac{\theta_z}{2}\right) . \end{aligned} \tag{7}$$

To check stability let $s_x = s_y = s_z = s$, and $\sin\theta_x/2 = \sin\theta_y/2 = \sin\theta_z/2 = 1$.

Then

$$G = 1 - \frac{12s}{(1+4\beta s)^3} \;. \tag{8}$$

so that the scheme is unconditionally stable if $\beta \ge 0.5$.

8.5 Modification of the program TWDIF to implement the ADI scheme (8.14 and 8.15) involves many changes. For example, the RHS is to be calculated twice and noting that (8.14 and 8.15) are written for T itself rather than the correction ΔT, the left hand side terms near the boundaries must be modified and the nature of the boundary conditions requires that they be imposed after every sweep.

A modified version of TWDIF is developed and the changes made are listed below. The modified program is written to be applicable to Problems 8.7 and 8.12 as well.

The computed solution errors for the ADI method for the test case of Table 8.2 are as follows.

$\Delta x, \Delta y =$	0.2	0.1	0.05
s = 0.5	0.02878	0.0066	0.00157
s = 1.0	0.0266	0.00648	0.00157
s = 5.0	0.01939	0.00244	0.00127

Thus we observe that the solution error, for a given value of s, is almost the same as for the approximate factorisation scheme (Table 8.2).

Considering their efficiencies, the ADI method involves the calculation of RHS twice and is less efficient for this problem.

Modifications to TWDIF for Problems 8.5, 8.7 and 8.12

Line 93, i.e. the call to REDIF is replaced by:

```
C
         SSY=1.-SY
         SYY=SY*0.5
C
         DO 221 J=2,NXN
         DO 221 K=2,NYN
C
C   FOR PROBLEMS 8.5, 8.7 AND 8.12
C
         R(K,J)=EMX(2)*(SYY*(T(K-1,J)+T(K+1,J))+SSY*T(K,J))
        1 + EMX(1)*(SYY*(T(K-1,J-1)+T(K+1,J-1))+SSY*T(K,J-1))
        2 + EMX(3)*(SYY*(T(K-1,J+1)+T(K+1,J+1))+SSY*T(K,J+1))
  221 CONTINUE
```

Lines 106 to 108 are replaced by

```
         IF(IBC .EQ. 1)RRT(JM) = RRT(JM) - B(4,JM)*T(K,NX)
```

Line 118 is replaced by

```
   20 T(K,J)= DDT(JM)
```

The following lines are inserted after line 119 of TWDIF.

```
C
C        TRIDIAGONAL SYSTEMS IN THE Y-DIRECTION
C
         SXX=0.5*SX
         SSX=1.-SX
         DO 222 J=2,NXN
         DO 222 K=2,NYN
C
C   FOR PROBLEMS 8.5, 8.7 AND 8.12
C
         R(K,J)=EMY(2)*(SXX*(T(K,J-1)+T(K,J+1))+SSX*T(K,J))
        1 + EMY(1)*(SXX*(T(K-1,J-1)+T(K-1,J+1))+SSX*T(K-1,J))
        2 + EMY(3)*(SXX*(T(K+1,J-1)+T(K+1,J+1))+SSX*T(K+1,J))
  222 CONTINUE
```

Line 131 is deleted and line 145 is replaced by

```
   23 T(K,J)= DDT(KM)
```

8.6 On running TWDIF for $s_x = s_y = 0.5$, 1.5 with Dirichlet boundary conditions (i.e. $IBC = 1$), the following solution errors are obtained for $t =$ 24,000 sec. and $\alpha = 1 \times 10^{-5}$.

s_x, s_y	$\Delta x, \Delta y$	Case 1	Case 2	Case 3	Case 4	Case 5
0.5	0.2	0.0288	0.2161	0.0234	0.0312	0.00059
	0.1	0.0066	0.0471	0.0063	0.0067	0.00003
	0.05	0.0016	0.011	0.00156	0.00158	0.000002
1.5	0.2	0.0234	0.5758	0.0304	0.0379	0.00661
	0.1	0.0063	0.1314	0.0035	0.0071	0.00036
	0.05	0.0016	0.0303	0.00139	0.00160	0.000021

The parameters for the various cases are given in the following table.

	Parameters
Case 1	ME=1, FDM, $\gamma = 0, \beta = 0.5$
Case 2	2LFI, FDM, $\gamma = 0, \beta = 1.0$
Case 3	3LFI, FDM, $\gamma = 0.5, \beta = 1.0$
Case 4	(Crank - Nicolson) ME =2, FEM, $\gamma = 0, \beta = 0.5$
Case 5	(Crank - Nicolson) ME=2, $\delta = 1/12, \gamma = 0, \beta = 0.5$

It is indicated in Sect. 8.3.2 that the Crank-Nicolson finite difference and finite element schemes and the $3LFI$ finite-difference schemes have second-order truncation errors, but the two-level scheme $2LFI$ has first-order truncation error. However, with $\delta = 1/12$, the Crank-Nicolson scheme has a fouth-order truncation error. The computed errors shown above for $s = 0.5$ and 1.5 clearly follow the truncation errors of the schemes. The schemes exhibit their second order accuracy (rate of convergence of 2) and as expected the FEM scheme with $\delta = 1/12$ shows a fourth-order accuracy. It is also seen that in general (with the exception of the Crank-Nicolson FDM scheme) the schemes have a lower accuracy when $s = 1.5$.

Comparing TWDIF with Scheme 8.7 (see Problem 8.3) we find that the former is more accurate, but the computed errors are of the same order of magnitude for a given value of s.

8.7 As pointed out earlier the program developed for Problem 8.5 has the option to implement the ADI-FEM scheme (8.35 and 8.36) and the parameter setting is ME =2 . Running the program with $\delta = 1/6$, the following solution errors are obtained. The test case is the one shown in Table 8.2.

$\Delta x, \Delta y =$	0.2	0.1	0.05
s=0.5	0.000098	0.0000054	0.00000032
s=1.0	0.004535	0.000088	0.00000519
s=5.0	0.03029	0.00354	0.000209

Comparing the accuracy with the ADI-FDM and the approximate factorisation schemes (Table 8.2), the present scheme exhibits superior accuracy. The rate of convergence is greater than 4 but errors grow with increasing s.

8.8 The scheme (8.43 and 8.44) with $\gamma = 0$ and $\beta = 0.5$ will reduce to

$$M_x \otimes M_y \frac{\Delta T_{j,k}^{n+1}}{\Delta t} = 0.5\alpha\Big(M_y \otimes L_{xx} + M_x \otimes L_{yy}\Big)\Big(T_{j,k}^n + T_{j,k}^{n+1}\Big). \qquad (1)$$

$$\text{i.e.} \quad \Big[M_x \otimes M_y - 0.5\alpha\Delta t(M_y \otimes L_{xx} + M_x \otimes L_{yy})\Big]T_{j,k}^{n+1}$$
$$= \Big[M_x \otimes M_y + 0.5\alpha\Delta t(M_y \otimes L_{xx} + M_x \otimes L_{yy})\Big]T_{j,k}^n . \qquad (2)$$

Following the ADI splitting as in (8.33) and (8.34),

$$\Big(M_x - 0.5\alpha\Delta t\ L_{xx}\Big)\Big(M_y - 0.5\alpha\Delta t\ L_{yy}\Big)T_{j,k}^{n+1}$$
$$= \Big(M_x + 0.5\alpha\Delta t\ L_{xx}\Big)\Big(M_y + 0.5\alpha\Delta t\ L_{yy}\Big)T_{j,k}^n . \qquad (3)$$

Now consider the operator $(M_x - 0.5\alpha\Delta t\ L_{xx})$,

$M_x = (\delta,\ 1 - 2\delta,\ \delta)$ so that $M_x T = (1 + \Delta x^2 \delta L_{xx})T$. (4)

Hence (3) can be written as

$$\Big[1 - (0.5\alpha\Delta t - \delta\Delta x^2)L_{xx}\Big]\Big[1 - (0.5\alpha\Delta t - \delta\Delta y^2)L_{yy}\Big]T_{j,k}^{n+1}$$
$$= \Big[1 + (0.5\alpha\Delta t + \delta\Delta x^2)L_{xx}\Big]\Big[1 + (0.5\alpha\Delta t + \delta\Delta y^2)L_{yy}\Big]T_{j,k}^n . \qquad (5)$$

which is equivalent to the scheme given in Problem 8.8. This can be checked by eliminating $T_{j,k}^*$ and writing the two as a single equation.

8.9 Modification required in the program TWDIF to replace $\Delta T_{NX,k}^*$ by an analytic solution at $t_{n+1/2}$, i.e. from (8.41), is again straightforward and is indicated in the following listing.

The solution errors computed when $s_x = s_y = 1$ are as follows (shown as Case A). Also given in the table are the errors when $\Delta T_{NX,k}^*$ is calculated analytically at t_{n+1} shown as Case B. In these computations the values of the various parameters were $t = 24,000$ sec., $\alpha = 1 \times 10^{-5}$, $\gamma = 0$, $\beta = 0.5$.

$\Delta x, \Delta y$	Case A	Case B
0.2	0.3556	0.0560
0.1	0.0877	0.00817
0.05	0.0221	0.00166

Comparing the errors with those in Table 8.2, we find that the scheme (8.9) is of lower accuracy. This is not unexpected in view of the discussion before (8.48).

Modifications to TWDIF for Problem 8.9

Insert after line 91:
```
      TFAC1 = EXP(DIM*(TIM+0.5*DTIM)) - EXP(DIM*TIM)
```

Line 108 is replaced by:
```
      IF(IBC .EQ. 1)RRT(JM) = RRT(JM) + B(4,JM)* 80.*SIY(K)*TFAC1
```

8.10 A closer look at TWDIF shows that when $IBC = 2$, the required Neumann conditions are imposed. The computed solution errors with $IBC = 2$ and $s_x = s_y = 0.5$, $t = 24,000$ sec., $\alpha = 1 \times 10^{-5}$ are as follows. The values of the parameters for different cases , like Case 1, Case 2 etc., are identical to those employed in Problem 8.6 and have been tabulated in the discussion of the same problem.

s_x, s_y	$\Delta x, \Delta y$	Case 1	Case 2	Case 3	Case 4	Case 5
0.5	0.2	0.1398	0.9319	0.116	0.1464	0.00282
	0.1	0.0353	0.2256	0.0312	0.033	0.00016
	0.05	0.0078	0.0545	0.00773	0.00783	0.000008
1.5	0.2	0.1119	0.0208	0.0683	0.1751	0.03112
	0.1	0.031	0.589	0.0185	0.0346	0.00178
	0.05	0.0077	0.1467	0.00696	0.00793	0.00011

The trends are similar to that observed in Table 8.3. The rms errors are now larger than when Dirichlet conditions are applied. The largest errors occur near the boundaries where Neumann conditions are imposed.

8.11 The program developed for Problem 8.3 has already an option to force Neumann boundary conditions (see listing). The computed errors for $s_x = s_y = 0.5$, $t = 24,000$ sec., $\alpha = 1 \times 10^{-5}$ are

$\Delta x, \Delta y =$	0.2	0.1	0.05
Error	0.2909	0.0658	0.01566

Comparing the errors with those for the Dirichlet boundary conditions (see Problem 8.3), we find that the implementation of the Neumann conditions causes the solution accuracy to deteriorate. The convergence rate is however preserved. Similar trends are observed with the implicit schemes (see Problem 8.10).

8.12 The program developed for Problem 8.5 has the option to impose the Neumann conditions at the specified boundaries (Table 8.3) and the parameter setting is IBC =2 . The solution errors computed with the Neumann conditions at $x = 1$ and $y = 1$ boundaries (for $s = 1.0$) are

$\Delta x, \Delta y =$	0.2	0.1	0.05
Error	0.1294	0.03199	0.0077

These errors are very close to those obtained with the approximate factorisation scheme (Table 8.3).

8.13 The 'ADI' program developed in Problem 8.5 is easily modified to implement the method of fractional steps (8.73 and 8.74). The basic changes consist of replacing the statements calculating the RHS and are indicated in the listing provided later.

The modified program (Case A) when run for the conditions of Table 8.2 produces the following solution errors. In the table the solution error given by TWDIF is shown as Case B. The following parameter values are set: $t = 24,000$ sec., $\alpha = 1 \times 10^{-5}$, $s_x = s_y = 1$, $\gamma = 0$, $\beta = 0.5$,

$\Delta x, \Delta y$	Case A	Case B
0.2	0.02666	0.02682
0.1	0.00648	0.00648
0.05	0.00157	0.00157

The solution error behaviour indicates that the scheme is second-order accurate for fixed s.

Modifications to the code used in Problem 8.5 for Problems 8.13 and 8.14 An integer $IPR14$ is read in and it acts as a switch between Problems 8.13 and 8.14. When $IPR14$ is 1 the scheme for Problem 8.14 is operative. An additional array BN of the same dimension as TN is defined. The lines before the statement- `DO 12 J= 1,NX` are modified as

```
      T(K,J) = 80.*(Y(K) - SIX(J)*SIY(K)) + 20.
      IF(IPR14 .EQ. 0) GO  TO 10
      IF(J .EQ. NX) BN(K)=T(K,J)
   10 CONTINUE
   11 CONTINUE
      IF(IBC .EQ. 1)GOTO 14
```

Lines between the statements `DO 221 J=2,NXN` and `GO TO 37` are replaced by

```
      DO 221 K=2,NYN
      R(K,J)=T(K,J)+0.5*SX*(T(K,J-1)+T(K,J+1)-2.*T(K,J))
  221 CONTINUE
C
C     TRIDIAGONAL SYSTEMS IN THE X-DIRECTION
C
      TIM = TIM + DTIM*0.5
      TFAC = EXP(DIM*TIM)
      IF(IBC .EQ. 2)GOTO 226
      DO 225 K = 1,NY
      T(K,NX) = 20. + 80.*(Y(K) - TFAC*SIY(K))
      IF(IPR14 .EQ. 1) BN(K)=T(K,NX)
  225 T(NY,K) = 20. + 80.*(1. - TFAC*SIY(K))
      GOTO 229
  226 DO 227 J = 1,NX
  227 T(NYS,J) = T(NYP,J) + 160.*DY
      DO 228 K = 1,NYS
  228 T(K,NXS) = T(K,NXP)
  229 CONTINUE
      DO 21 K = 2,NYN
      DO 19 J = 2,NXN
      JM = J - 1
      B(2,JM) = EMX(1) - CCXA
      B(3,JM) = EMX(2) + 2.*CCXA
      B(4,JM) = EMX(3) - CCXA
   19 RRT(JM) = R(K,J)
      IF(IBC .EQ. 1)DT(K,NX) = - 80.*SIY(K)*TFAC
      IF(IBC .EQ. 1)RRT(JM) = RRT(JM) - B(4,JM)*T(K,NX)
      IF(IBC .EQ. 1 .OR. IBC .EQ. 2)RRT(1) =
     1  RRT(1) - B(2,1)*T(K,1)
      IF(IBC .EQ. 2)DT(K,NXS) = DT(K,NXP)
      IF(IBC .EQ. 2)B(2,JM) = B(2,JM) + B(4,JM)
      B(4,JM) = 0.
c     IF(IBC .EQ. 1) B(2,1)=0.0
      B(2,1)=0.0
C
      CALL BANFAC(B,NXPP,1)
      CALL BANSOL(RRT,DDT,B,NXPP,1)
C
      DO 20 J = 2,NXN
      JM = J - 1
   20 T(K,J)= DDT(JM)
   21 CONTINUE
C
C     TRIDIAGONAL SYSTEMS IN THE Y-DIRECTION
C
      DO 222 J=2,NXN
      DO 222 K=2,NYN
      R(K,J)=T(K,J)+0.5*SY*(T(K-1,J)+T(K+1,J)-2.*T(K,J))
      IF(IPR14 .EQ. 1. AND. J.EQ.NXN) THEN
      R(K,J)=BN(K)+0.5*SY*(BN(K-1)+BN(K+1)-2.*BN(K))
      ENDIF
  222 CONTINUE
C
      IF(IBC .EQ. 1)DT(NY,NX) = - 80.*TFAC
      IF(IBC .EQ. 2)DT(NYS,NXS) = DT(NYP,NXP)
      TIM = TIM + DTIM*0.5
      TFAC = EXP(DIM*TIM)
      IF(IBC .EQ. 2)GOTO 26
      DO 25 K = 1,NY
      T(K,NX) = 20. + 80.*(Y(K) - TFAC*SIY(K))
      IF(IPR14 .EQ. 1) BN(K)=T(K,NX)
   25 T(NY,K) = 20. + 80.*(1. - TFAC*SIY(K))
      GOTO 29
   26 DO 27 J = 1,NX
   27 T(NYS,J) = T(NYP,J) + 160.*DY
      DO 28 K = 1,NYS
   28 T(K,NXS) = T(K,NXP)
   29 CONTINUE
      NN=NX
      IF(IPR14 .EQ. 0) NN=NXN
      DO 24 J = 2,NN
      DO 22 K = 2,NYN
      KM = K - 1
      B 2,KM) = EMY(1) - CCYA
      B(3,KM) = EMY(2) + 2.*CCYA
      B(4,KM) = EMY(3) - CCYA
   22 RRT(KM) = R(K,J)
      IF(IBC .EQ. 1)DT(NY,J) = - 80.*SIX(J)*TFAC
      IF(IBC .EQ. 1)RRT(KM) = RRT(KM) - B(4,KM)* T(NY,J)
      IF(IBC .EQ. 1 .OR. IBC .EQ. 2)RRT(1) =
     1  RRT(1) - B(2,1)* T(1,J)
      IF(IBC .EQ. 2)DT(NYS,J) = DT(NYP,J)
```

```
      IF(IBC .EQ. 2)B(2,KM) = B(2,KM) + B(4,KM)
      IF(IBC .EQ. 2) RRT(KM) = RRT(KM)-B(4,KM)*160.*DY
      B(4,KM) = 0.
      B(2,1)=0.0
C
      CALL BANFAC(B,NYPP,1)
      CALL BANSOL(RRT,DDT,B,NYPP,1)
C
      INCREMENT T
C
      DO 23 K = 2,NYN
      KM = K - 1
      DT(K,J) = DDT(KM)
   23 T(K,J) =  DDT(KM)
   24 CONTINUE
      GAM = GAMH
      BET = BETH
      IF(TIM .GE. TMCH)GOTO 30
      IF(TIM .LE. TMAX)GOTO 18
      GOTO 37
```

8.14 The code developed for Problem 8.14 has the option to implement the Dirichlet condition on x=1, i.e. $IPR14$=1 . On running the modified program the following rms solution errors were obtained.

$\Delta x, \Delta y$	solution error
0.2	0.02877
0.1	0.00661
0.05	0.00157

The solution errors are very close to those observed for the conventional method of fractional steps.

CTFD Solutions Manual: Chapter 9

Linear Convection-Dominated Problems

9.1 Here we consider the truncation error for the Lax-Wendroff scheme for the convection equation. The governing equation is

$$\frac{\partial T}{\partial t} + u\frac{\partial T}{\partial x} = 0 \,, \tag{1}$$

and the Lax-Wendroff scheme is given by

$$\frac{\Delta T_j^{n+1}}{\Delta t} + uL_xT_j^n - 0.5uC\Delta x\ L_{xx}T_j^n = 0. \tag{2}$$

Expanding every term in (2) as a Taylor series about T_j^n and rearranging the terms we have

$$\begin{aligned} T_t + uT_x + \frac{\Delta t}{2}T_{tt} - \frac{u^2\Delta t}{2}T_{xx} + \frac{\Delta t^2}{6}T_{t3} + \frac{u\Delta x^2}{6}T_{x3} \\ + \frac{\Delta t^3}{24}T_{t4} - \frac{u^2}{24}\Delta t\Delta x^2\ T_{x4} + O(H) = 0\ . \end{aligned} \tag{3}$$

The modified equation is obtained by eliminating the time derivatives T_{tt}, T_{t3} etc. Recall in Chapter 4 the governing equation itself was used to eliminate the time derivatives, thus arriving at an approximate expression for the truncation error. In order to obtain an exact expression for the truncation error, equation (3) which is the actual differential equation solved as a consequence of discretisation, should be used to replace the time derivatives.

To carry out this process of elimination it is useful to construct a table of coefficients (see Table 1). The technique is self-explanatory. The coefficients of terms of the modified equation are obtained by adding the terms in each column. Thus, we have

$$T_t + uT_x + \frac{u}{6}(\Delta x^2 - u^2\Delta t^2)T_{x3} + \frac{u^2\Delta t}{8}(\Delta x^2 - u^2\Delta t^2) + O(H) = 0\ . \tag{4}$$

Noting that $u\Delta t/\Delta x = C$, (4) reduces to

$$T_t + uT_x + \frac{u\ \Delta x^2}{6}(1 - C^2)T_{x3} + \frac{uC}{8}\Delta x^3(1 - C^2)T_{x4} + O(H) = 0\ ,$$

as required.

Table 1. Derivation of the modified equation for the Lax-Wendroff scheme.

	T_t	T_x	T_{tt}	T_{tx}	T_{xx}	T_{ttt}	T_{ttx}	T_{txx}	T_{xxx}
Coefficients of (3)	1	u	$\Delta t/2$	0	$-u^2\Delta t/2$	$\Delta t^2/6$	0	0	$u\Delta x^2/6$
$-\Delta t/2\ \partial/\partial t$ (3)			$-\Delta t/2$	$-u\Delta t/2$	0	$-\Delta t^2/4$	0	$u^2\Delta t^2/4$	0
$u\Delta t/2\ \partial/\partial x$ (3)				$u\Delta t/2$	$u^2\Delta t/2$	0	$u\Delta t^2/4$	0	$-u^3\Delta t^2/4$
$\Delta t^2/12\ \partial^2/\partial t^2$ (3)						$\Delta t^2/12$	$u\Delta t^2/12$	0	0
$-u/3\ \Delta t^2\ \partial^2/\partial t\partial x$ (3)							$-u\Delta t^2/3$	$-u^2\Delta t^2/3$	0
$u^2/12\ \Delta t^2\ \partial^2/\partial x^2$ (3)								$u^2\Delta t^2/12$	$u^3\Delta t^2/12$
$u/12\ \Delta t^3\ \partial^3/\partial t^2\partial x$ (3)									
$-u^2/12\ \Delta t^3\ \partial^3/\partial t\partial x^2$ (3)									
$u\Delta t/12\ (\Delta x^2 - u^2\Delta t^2)\ \partial^3/\partial x^3$ (3)									
Sum of coefficients	1	u							$u/6\ (\Delta x^2 - u^2\Delta t^2)$

Table 1. continued.

	T_{tttt}	T_{tttx}	T_{ttxx}	T_{txxx}	T_{xxxx}
Coefficients of (3)	$\Delta t^3/24$	0	0	0	$-u^2\Delta t\Delta x^2/24$
$-\Delta t/2\ \partial/\partial t$ (3)	$-\Delta t^3/12$	0	0	$-u\Delta t\Delta x^2/12$	0
$u\Delta t/2\ \partial/\partial x$ (3)	0	$u\Delta t^3/12$	0	0	$u^2\Delta t\Delta x^2/12$
$\Delta t^2/12\ \partial^2/\partial t^2$ (3)	$\Delta t^3/24$	0	$-u^2\Delta t^3/24$	0	0
$-u/3\ \Delta t^2\ \partial^2/\partial t\partial x$ (3)	0	$-u\Delta t^3/6$	0	$u^3\Delta t^3/6$	0
$u^2/12\ \Delta t^2\ \partial^2/\partial x^2$ (3)	0	0	$u^2\Delta t^3/24$	0	$-u^4\Delta t^3/24$
$u/12\ \Delta t^3\ \partial^3/\partial t^2\partial x$ (3)		$u\Delta t^3/12$	$u^2\Delta t^3/12$	0	0
$-u^2/12\ \Delta t^3\ \partial^3/\partial t\partial x^2$ (3)			$-u^2\Delta t^3/12$	$-u^3\Delta t^3/12$	0
$u\Delta t/12\ (\Delta x^2-u^2\Delta t^2)\ \partial^3/\partial x^3$ (3)				$u\Delta t/12\ (\Delta x^2-u^2\Delta t^2)$	$u^2\Delta t/12\ (\Delta x^2-u^2\Delta t^2)$
Sum of coefficients	0	0	0	0	$u^2\Delta t/8\ (\Delta x^2-u^2\Delta t^2)$

9.2 The von Neumann stability analysis of the Crank-Nicolson mass operator scheme is carried out as follows.

The given scheme is

$$(\delta-0.25C)T_{j-1}^{n+1}+(1-2\delta)T_j^{n+1}+(\delta+0.25C)T_{j+1}^{n+1}=$$
$$(\delta+0.25C)T_{j-1}^{n}+(1-2\delta)T_j^{n}+(\delta-0.25C)T_{j+1}^{n}\ . \tag{1}$$

Following the procedure outlined before, we have the amplification factor

$$G=\frac{1-2\delta+2\delta\cos\theta-0.5\ iC\sin\theta}{1-2\delta+2\delta\ \ \cos\theta+0.5\ iC\sin\theta}\ . \tag{2}$$

Carrying out the analysis, we find that $|G|\le 1.0$ for all values of C and δ i.e. scheme (1) is unconditionally stable.

9.3 Here the Crank-Nicolson mass operator scheme is applied to the problem of a convecting sine wave.

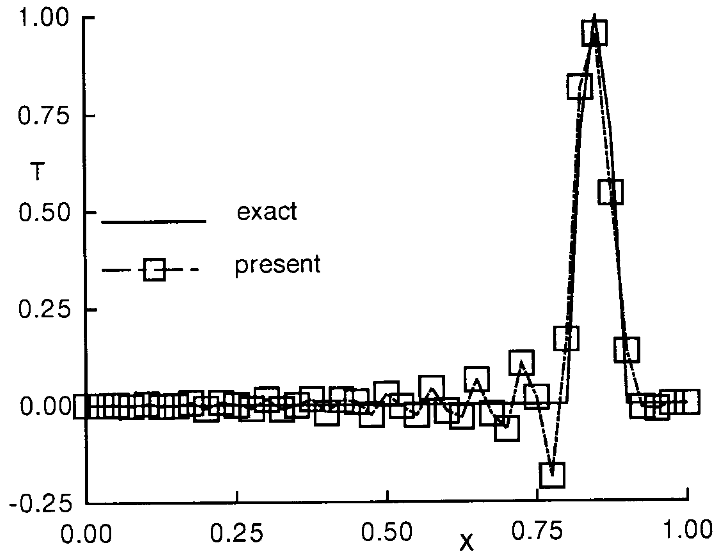

Fig. 1. Temperature profiles at t=9 sec.

For this application the program TRAN is used with $ME=4$ and $JPR=1$. Further, the mass operator δ is given by $(1/6\ +C^2/12)$ (see Table 9.4) where C is the Courant number. Also $\alpha=s=0$.

The computed temperature profile at $t=9$ sec. is shown in Fig.1. In comparison with results shown in Figs. 9.2, 9.3 and 9.4 we observe that the dissipative and dispersive errors are drastically reduced, thus indicating that the Crank-Nicolson mass operator scheme is very accurate.

9.4 Fourier analysis is applied to the one-dimensional convection equation. Introducing the Fourier representation for the computational solution in the form

$$T(x,t) = \sum_{m=-\infty}^{\infty} T_m\, e^{-p(m)t} e^{im[x-q(m)t]} \ , \tag{1}$$

to each of the schemes indicated, we get after simplification:

i) Upwind difference scheme:

$$T_j^{n+1} = (1-C)T_j^n + CT_{j-1}^n \ , \tag{2}$$

$$G_m = 1 + C\big[\cos(m\Delta x) - 1\big] + i\ C\ \sin(m\Delta x) \ , \tag{3}$$

$$\begin{aligned} |G|_m &= \big((1 + C\ \cos(m\Delta x) - C)^2 + C^2\ \sin^2(m\Delta x)\big)^{\frac{1}{2}} \\ &= \big(1 - 4C(1-C)\sin^2(m\Delta x/2)\big)^{\frac{1}{2}} \ , \end{aligned} \tag{4}$$

$$\phi_m = tan^{-1}\frac{-C\ \sin(m\Delta x)}{1 + C[\cos(m\Delta x) - 1]} \ .$$

ii) Lax-Wendroff scheme:

$$T_j^{n+1} = 0.5C(T_{j+1}^n - T_{j-1}^n) + 0.5C^2(T_{j-1}^n - 2T_j^n - T_{j+1}^n) \ , \tag{5}$$

$$G_m = 1 - 2C^2 \sin^2(m\Delta x/2) - iC\ \sin(m\Delta x) \ , \tag{6}$$

$$|G|_m = (1 + \sin^4(m\Delta x/2)\,4C^2(C^2-1))^{\frac{1}{2}} \ , \tag{7}$$

$$\phi_m = tan^{-1}\frac{-C\ \sin(m\Delta x)}{1 - 2C^2\sin^2(m\Delta x/2)} \ . \tag{8}$$

iii) Crank-Nicolson scheme:

$$-0.25T_{j-1}^{n+1} + T_j^{n+1} + 0.25T_{j+1}^{n+1} = 0.25CT_{j-1}^n + T_j^n - 0.25CT_{j+1}^n \ . \tag{9}$$

Carrying out the analysis as before,

$$G = \frac{1 - 0.5\ i\ C\ \sin(m\Delta x)}{1 + 0.5\ i\ C\ \sin(m\Delta x)} \ , \tag{10}$$

$$|G_m| = 1.0 \ , \tag{11}$$

$$\phi_m = \frac{-C\ \sin(m\Delta x)}{1 - (0.5\ C\ \sin(m\Delta x))^2} \ . \tag{12}$$

On calculating $|G|_m$ and ϕ_m for various schemes and various values of $m\Delta x (= \theta_m)$, we have for $C = 0.8$, $\Delta x = 0.5$

Scheme	$\theta_m/\pi =$	0.05	0.25	0.5	0.75	1.0
Upwind	G_m/G_{ex}	0.998	0.952	0.825	0.674	0.6
Lax-Wen.	G_m/G_{ex}	1.0	0.99	0.877	0.573	0.280
CN	G_m/G_{ex}	1.0	1.0	1.0	1.0	1.0
exact	$\phi_{ex,m}$	-9.0	-45.0	-90.0	-135.0	-180.0
Upwind	ϕ_m	-7.2	-36.46	-75.96	-122.88	-180
Lax-Wen.	ϕ_m	-7.19	-34.85	-65.77	-99.29	-180
CN	ϕ_m	-7.16	-31.59	-43.6	-31.59	0

The upwind scheme has greater 'dissipation error' for $\theta_m/\pi = 0.25$ and 0.5 when compared with the Lax-Wendroff scheme. This trend is reversed when θ_m/π is 0.75 and 1.0. The phase error is always larger for the Lax-Wendroff scheme. These features explain the smooth but highly diffused profile obtained with the upwind scheme. The Lax-Wendroff scheme, though having a lower overall dissipation error, has substantial phase error explaining the larger amplitude of the primary wave (Fig. 9.3) which lags the theoretical. The Crank-Nicolson scheme has no dissipation error but has substantial phase errors, explaining the highly oscillatory profile (Fig. 9.4).

9.5 Here the modified equation is applied to the upwind difference scheme, Lax-Wendroff scheme and Crank-Nicolson finite-difference scheme.

Following the procedure of Problem 9.1, we have:

1) Upwind Difference scheme:

$$\frac{\Delta T_j^{n+1}}{\Delta t} + u\frac{T_j^n - T_{j-1}^n}{\Delta x} = 0 \ . \tag{1}$$

A Taylor series expansion of the terms in the above equation gives,

$$\begin{aligned} T_t + uT_x + \frac{\Delta t}{2}T_{tt} - \frac{u\Delta x}{2}T_{xx} + \frac{\Delta t^2}{6}T_{t^3} + \frac{u\Delta x^2}{6}T_{x^3} + \\ \frac{\Delta t^3}{24}T_{t^4} - \frac{u\Delta x^3}{24}T_{x^4} + O(H) = 0 \ . \end{aligned} \tag{2}$$

Constructing the table and eliminating the time derivatives, we have,

$$\begin{aligned} \frac{\partial T}{\partial t} + u\frac{\partial T}{\partial x} - \frac{u\Delta x}{2}(1-C)\frac{\partial^2 T}{\partial x^2} + \frac{u\Delta x^2}{6}(1 - 3C + 2C^2)\frac{\partial^3 T}{\partial x^3} \\ + \frac{u\Delta x^3}{24}(6C^3 - 12C^2 + 7C - 1)\frac{\partial^4 T}{\partial x^4} + O(H) = 0 \ . \end{aligned} \tag{3}$$

ii) Lax-Wendroff scheme:

The modified equation has already been derived in Problem 9.1.

iii) Crank-Nicolson finite difference scheme:

$$\frac{\Delta T_j^{n+1}}{\Delta t} + uL_x\left(\frac{T_j^n + T_j^{n+1}}{2}\right) = 0\ . \tag{4}$$

A Taylor series expansion of the terms in (4) gives,

$$\begin{aligned} T_t + uT_x + \frac{\Delta t}{2}T_{tt} + \frac{u}{2}\Delta t\ T_{tx} + \frac{\Delta t^2}{6}T_{t^3} \\ + \frac{u\Delta t^2}{4}T_{ttx} + \frac{u}{6}\Delta x^2\ T_{x^3} + \frac{\Delta t^3}{24}T_{t^4} + \frac{u\Delta t^3}{12}T_{t^3x} \\ + \frac{u}{12}\Delta t\ \Delta x^2\ T_{tx^3} + \frac{u\Delta t^4}{48}T_{t^4x} + O(H) = 0\ . \end{aligned} \tag{5}$$

Constructing the table and eliminating the time derivatives as before, we have

$$T_t + uT_x + \left(\frac{u^3\ \Delta t^2}{12} + \frac{u}{6}\Delta x^2\right)T_{x^3} + O(H) = 0\ , \quad \text{or}$$

$$T_t + uT_x + \frac{u\Delta x^2}{6}\left(\frac{C^2}{2} + 1\right)T_{x^3} + O(H) = 0\ .$$

Thus the results shown in Table 9.1 are confirmed.

Discussion:

The upwind difference scheme has a first-order dissipation and a second-order dispersion error. The Lax-Wendroff scheme has a second order dispersion and third order dissipation error. Thus one expects a larger amplitude (ie. lesser diffusion of the initial sine wave) for the Lax-Wendroff scheme. For the upwind scheme, the larger dissipation error is the dominant one, thus producing a 'clean' but highly damped profile. For the Crank-Nicolson scheme, the dissipation errors are of magnitude third-order or higher while there is a second-order dispersion error. This explains the highly oscillatory solution obtained.

9.6 The construction of a generalised, three level scheme for the one-dimensional equation can be implemented as follows.

The general three-level scheme is given by

$$\frac{(1+\gamma)\Delta T^{n+1} - \gamma\Delta T^n}{\Delta t} + u(1-\beta)L_xT_j^n + u\beta L_xT_j^{n+1} = 0\ . \tag{1}$$

Expanding every term of (1) in a Taylor series about T_j^n and simplifying we have,

$$\begin{aligned} T_t + uT_x + \Delta t\left(\frac{1}{2}+\gamma\right)T_{tt} + u\beta\Delta t\ T_{tx} + \frac{\Delta t^2}{6}T_{ttt} + \frac{u\beta\Delta t^2}{6}T_{ttx} \\ + \frac{u\Delta x^2}{6}T_{xxx} + (\frac{1}{2}+\gamma)\frac{\Delta t^3}{12}T_{t^4} + \frac{u\beta\Delta t^3}{6}T_{t^3x} \\ + \frac{u\beta\ \Delta t\ \Delta x^2}{6}T_{tx^3} + O(H) = 0\ . \end{aligned} \tag{2}$$

Replacing the time derivatives in (2) by space derivatives, using,

$T_t = -uT_x$, $T_{tx} = -uT_{xx}$, $T_{tt} = u^2T_{xx}$, $T_{t^2x} = u^2T_{x^3}$, $T_{t^3} = -u^3T_{x^3}$, $T_{t^3x} = -u^3\ T_{x^4}$, $T_{t^4} = u^4\ T_{x^4}$ etc, we have

$$\begin{aligned} T_t + uT_x + u^2\ \Delta t\{(0.5+\gamma) - \beta\}T_{xx} + \Delta t^2u^3(-\frac{1}{6} + 0.5\beta)T_{xxx} \\ + O(H) = 0\ . \end{aligned} \tag{3}$$

A von Neumann analysis of (1) gives the following expression for the amplification factor G,

$$G^2[(1+\gamma) + iC\beta\sin\theta] - G[(1+2\gamma) + iC(1-\beta)\sin\theta] + \gamma = 0\ . \tag{4}$$

The scheme is found to be unconditionally stable when $\beta = 1/2 + \gamma$.

Computational experiment: The program TRAN has been modified to incorporate the above three-level discretisation. The important changes are in the calculation of the left and the right hand sides of the tridiagonal equation which now becomes

$$\begin{aligned} -0.5C\beta\ T_{j-1}^{n+1} + (1+\gamma)T_j^{n+1} + 0.5\ C\beta\ T_{j+1}^{n+1} \\ = 0.5C(1-\beta)T_{j-1}^n + (1+2\gamma)T_j^n - 0.5\ C(1-\beta)T_{j+1}^n - \gamma T_j^{n-1}. \end{aligned} \tag{5}$$

The listing provided below shows the parts of the code undergoing important changes.

The modified program was run for the case of a propagating sine wave. The truncation error analysis indicates that for minimum dissipation and dispersion errors $\beta = 1/3$ and $\gamma \le -1/6$. This choice of β and γ gives an oscillatory profile since there is no physical dissipation in the problem. To provide some dissipation for stability purposes β was set to be 0.4. The resulting temperature profile is shown in Fig. 1.

Modifications to TRAN for Problem 9.6

1. In the READ statement (line 10) variables TST, BET and GAM are read in instead of S, Q and EM.
2. Lines 25 to 31, and 54 to 60 are deleted. Lines 61 to 66 are replaced by,

```
C
   12 AA = -0.5*C*BET
      BB = 1.0 + GAM
      CC = -AA
```

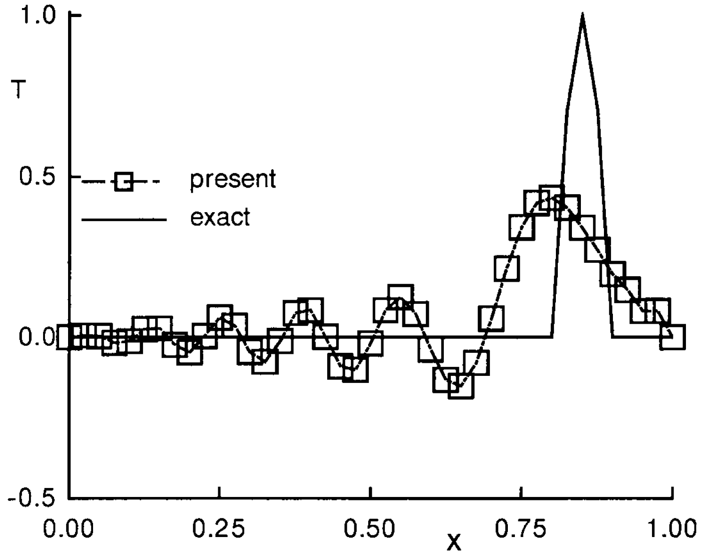

Fig. 1. Temperature profile for the propagating sine wave given by the modified scheme.

```
      AE = 0.5*C*(1. -BET)
      BE = 1. + 2.*GAM
      CE = -AE
C
```

3. The following lines are inserted after line 87:

```
      TIM = TST - 2.*DT
      DO 110 I = 1,2
      TIM = TIM + DT
C
      CALL EXSOL(JPR,JMAX,X,T,TEX,NEX,DX,U,ALPH,TIM,EL)
C
      DO 99 J = 2,JMAP
      IF(I .EQ. 1)TOL(J) = TEX(J)
   99 CONTINUE
  110 CONTINUE
```

4. Lines 89 to 138 are replaced by:

```
      DO 26 N = 1,NTIM
C
C     TRIDIAGONAL SYSTEM FOR IMPLICIT SCHEMES
C
      TIM = TIM + DT
```

```
   21 CONTINUE
      DO 22 J = 2,JMAP
      JM = J - 1
      A(1,JM) = 0.0
      A(2,JM) = AA
      A(3,JM) = BB
      A(4,JM) = CC
      R(JM) = AE*T(JM) + BE*T(J) + CE*T(J+1) - GAM*TOL(J)
   22 CONTINUE
      R(1) = R(1) - A(2,1)*T(1)
   24 A(2,1) = 0.
      A(4,JMAF) = 0.
```

9.7 For the steady convection-diffusion equation, the centred and the upwind difference schemes are given by

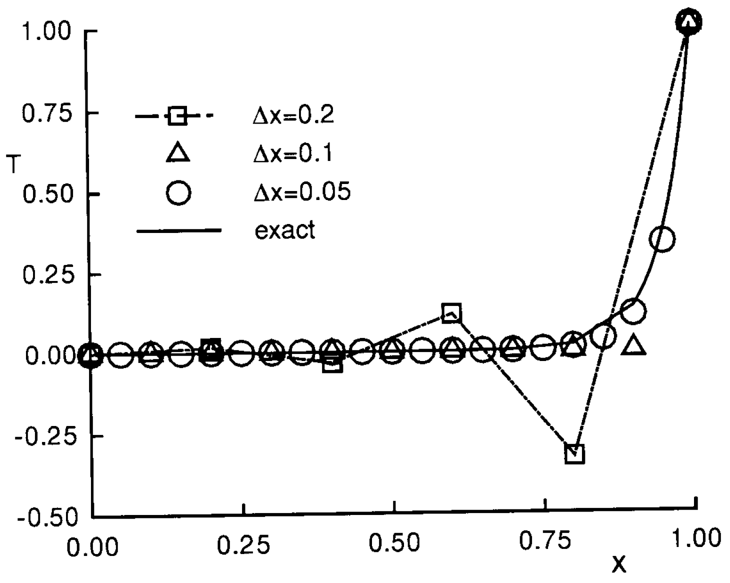

Fig. 1. Temperature profile given by centred difference scheme.

$$-(1 + 0.5\ R_{cell})T_{j-1} + 2T_j - (1 - 0.5\ R_{cell})T_{j+1} = 0\ , \tag{1}$$

and

$$-(1 + R_{cell})T_{j-1} + 2(1 + 0.5\ R_{cell})T_j - T_{j+1} = 0\ . \tag{2}$$

Any of the programs like TRAN can be modified to apply these schemes. The major change in the program involves:

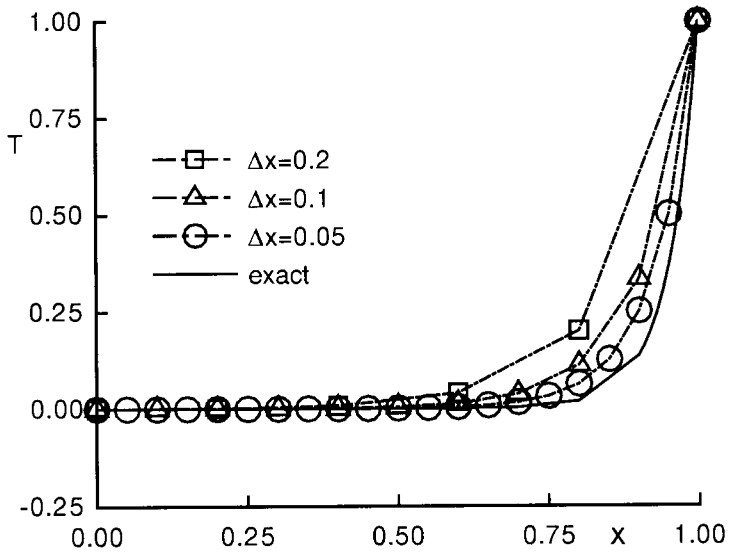

Fig. 2. Temperature profile given by upwind difference scheme.

Variable	Centred Scheme	Upwind Scheme
$AD =$	$-(1 + 0.5{*}RCELL)$	$-(1 + RCELL)$
$CD =$	$-(1 - 0.5{*}RCELL)$	-1
$BD =$	2	$2{*}(1 + 0.5{*}\ RCELL)$

The resulting temperature distribution are shown in the Figs. 1 and 2 (for $u/\alpha = 20$).

9.8 Neumann boundary conditions for the steady convective-diffusion equation are introduced in this problem.

For the exact solution given by (9.44), the temperature gradient g at the exit (i.e. $x = 1$) is

$$g = \left(\frac{u/\alpha\ e^{u/\alpha}}{e^{u/\alpha} - 1}\right). \tag{1}$$

The program developed for Problem 9.7 is modified to force the Neumann condition at the exit.

For centred differencing, imposition of Neumann condition at the exit gives,

$$\frac{T_{NX+1} - T_{NX-1}}{2\Delta x} \quad = \quad g \tag{2}$$

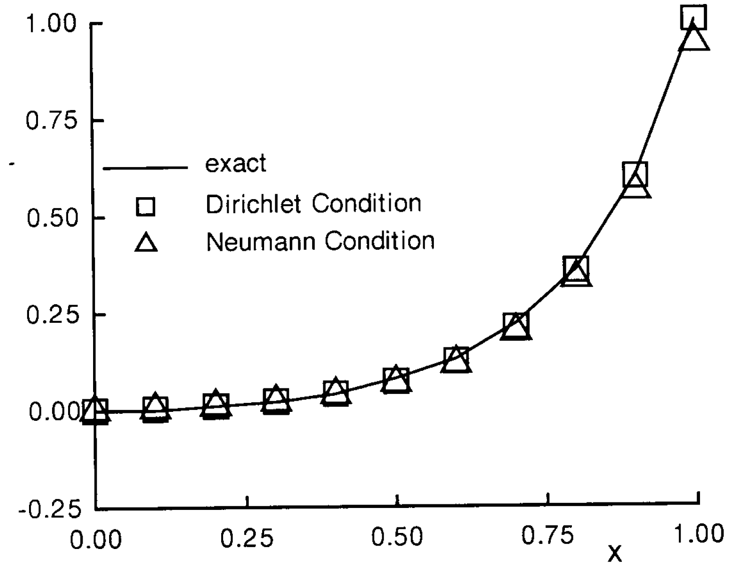

Fig. 1. Temperature profile given by centred difference scheme.

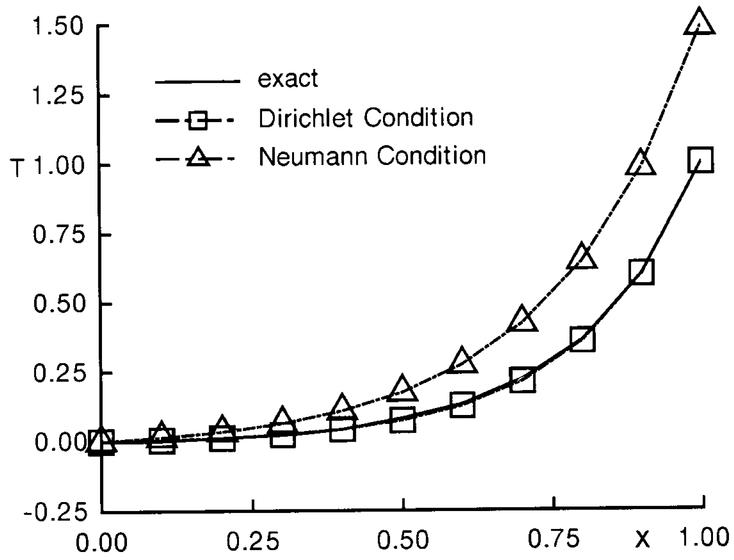

Fig. 2. Temperature profile given by upwind difference scheme.

and the algorithm for the steady convection-diffusion equation at the exit is

$$u\frac{T_{NX+1}-T_{NX-1}}{2\Delta x}-\alpha\frac{T_{NX+1}-2T_{NX}+T_{NX-1}}{\Delta x^2}=0\ . \tag{3}$$

Substituting (2) in (3) and simplifying, we have

$$T_{NX-1}-T_{NX}=g\Delta x(\frac{1}{2}R_{cell}-1)\ . \tag{4}$$

For the two-point upwind scheme, we have at exit,

$$-T_{NX-1}+T_{NX}=g\Delta x\ . \tag{5}$$

The results obtained on running the modified program are shown in Figs. 1 and 2 (where $u/\alpha=5.0$).

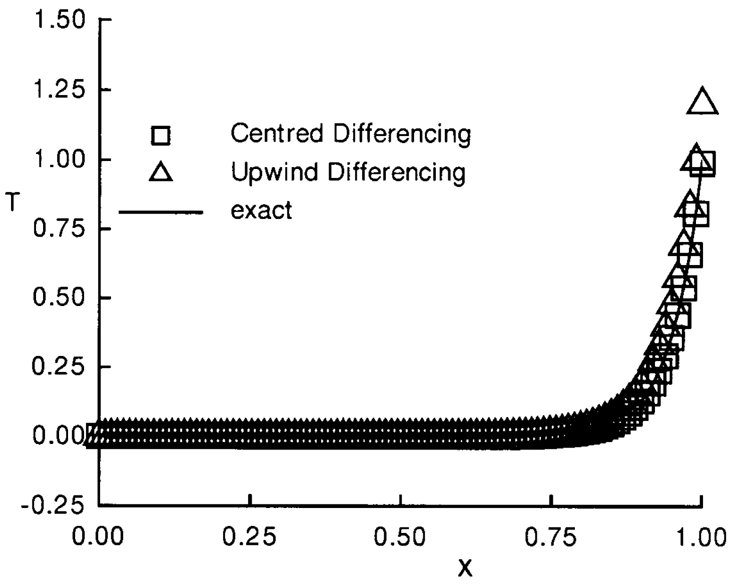

Fig. 3. Temperature profile given by different schemes for $\Delta x=0.01$.

With centred differencing the computed temperature profile follows the exact trend. But at $x=1$ the temperature value is underpredicted. This is a direct consequence of the fact that a gradient is prescribed at $x=1$. With upwind differencing the accuracy is inferior with an overprediction of temperature at $x=1$.

A sufficiently fine mesh should be able to give a more satisfactory agreement with the exact solution. That this is in fact true is exhibited when a fine mesh ($\Delta x=0.01$) is employed (Fig. 3). For this computation a u/α of 20 was used.

For this value of u/α, coarse grids give a solution which is quite inaccurate. But with a fine grid the agreement is good.

9.9 This problem considers five-point discretisation of first and second spatial derivatives

$$\text{Let}\quad \frac{d\bar{T}}{dx}=\frac{1}{\Delta x}\{c(T_{j+2}-T_{j-2})+b(T_{j+1}-T_{j-1})+aT_j\}\ . \tag{1}$$

Expanding the terms as a Taylor series about T_j, we have

$$\frac{d\bar{T}}{dx}=\frac{1}{\Delta x}\{aT+T_x\Delta x(4c+2b)+\frac{\Delta x^3}{6}T_{x^3}(16c+2b)\}+O(H)\ , \tag{2}$$

so that $a=0$, $4c+2b=1$ or $b+2c=1/2$. (3)

Similarly let

$$\frac{d\bar{T}}{dx}=\frac{1}{\Delta x^2}\{e(T_{j+2}+T_{j-2})+f(T_{j+1}+T_{j-1})+gT_j\} \tag{4}$$

$$\text{and}\quad \begin{aligned}\frac{d\bar{T}}{dx}=&\frac{1}{\Delta x^2}\Big[(2e+2f+g)T+\frac{\Delta x^2}{2}T_{xx}(8e+2f)\\&+\frac{\Delta x^4}{24}T_{x^4}(32e+2f)\Big]+O(H)\ ,\end{aligned} \tag{5}$$

so that $2e+2f+g=0$, $8e+2f=2$ or $4e+f=1$. (6)

The scheme to be developed, therefore, has the form

$$\begin{aligned}&\frac{u}{\Delta x}\left[c(T_{j+2}-T_{j-2})+b(T_{j+1}-T_{j-1})+aT_j\right]\\&-\frac{\alpha}{\Delta x^2}\left[e(T_{j+2}+T_{j-2})+f(T_{j+1}+T_{j-1})+gT_j\right]=0\ .\end{aligned} \tag{7}$$

Substituting expressions (4) and (5) in (7) and simplifying, we have

$$\begin{aligned}&uT_x-\alpha T_{xx}\\&=-\frac{u\Delta x^3}{6}T_{x^3}(16c+2b)+\frac{\alpha\ \Delta x^4}{24}T_{x^4}(32e+2f)+O(H)=0\ .\end{aligned} \tag{8}$$

The truncation error is minimised when $16c+2b=0$ and $32e+2f=0$.

Consequently, $c=-1/12$, $b=2/3$, $a=0$, $e=-1/12$, $f=4/3$ and $g=-5/2$ and the scheme is

$$\begin{aligned}&\frac{u}{\Delta x}\left[-\frac{T_{j+2}-T_{j-2}}{12}+\frac{2}{3}(T_{j+1}-T_{j-1})\right]\\&-\frac{\alpha}{\Delta x^2}\left[-\frac{T_{j+2}+T_{j-2}}{12}+\frac{4(T_{j+1}+T_{j-1})}{3}-\frac{5}{2}T_j\right]=0\ .\end{aligned}$$

Substituting R_{cell} for $u\Delta x/\alpha$, and simplifying, we get

$$(1+R_{cell})T_{j-2}+(-8\,R_{cell}-16)T_{j-1}+30T_j+(8R_{cell}-16)T_{j+1} \\ +(-R_{cell}+1)T_{j+2}=0\ , \tag{9}$$

which is a pentadiagonal system.

(It may be noted that the coefficients a, b, c d, ... are identical to the ones given in Tables 3.1 and 3.2).

Writing a program to invert (9) and running it, the following solution errors are obtained. The solution errors given by the centred three point differences in Problem 9.7 are also given for comparison.

Δx	Problem 9.9	Problem 9.7
0.1	0.196×10^{-1}	0.412×10^{-1}
0.05	0.926×10^{-3}	0.973×10^{-2}

Thus the five-point scheme (Problem 9.9) is seen to be more accurate than the centred three-point scheme (9.7), particularly as the grid is refined.

9.10 a) A modified equation for the Lax-Wendroff scheme is applied to the one-dimensional transport equation.

The scheme is

$$\frac{T_j^{n+1}-T_j^n}{\Delta t}+u\frac{T_{j+1}^n-T_{j-1}^n}{2\Delta x}-\alpha^*\left(\frac{T_{j+1}^n-2T_j^n+T_{j-1}^n}{\Delta x^2}\right)=0\ , \tag{1}$$

$$\text{with } \alpha^*=\alpha+0.5\ uC\Delta x\ . \tag{2}$$

Expanding every term of (1) in a Taylor series about T_j^n and simplifying we have

$$T_t+uT_x-\alpha^*T_{xx}+\frac{\Delta t}{2}T_{tt}+\frac{\Delta t^2}{6}T_{t^3}+\frac{\Delta t^3}{24}T_{t^4}+\frac{\Delta x^2}{6}uT_{x^3} \\ +\frac{\Delta x^4}{120}uT_{x^5}-\alpha^*\left(\frac{\Delta x^2}{12}T_{x^4}\right)-\alpha^*\left(\frac{\Delta x^4}{360}T_{x^6}\right)+O(H)=0\ . \tag{3}$$

Constructing the tables and eliminating the time derivatives,

$$T_t+uT_x-\alpha T_{xx}-\left\{C\alpha\Delta x-\frac{u\Delta x^2}{6}(1-C^2)\right\}T_{x^3} \\ +\left\{\frac{C\alpha^2\Delta x}{2u}-\frac{\alpha\Delta x^2}{12}-\frac{uC\Delta x^3}{8}(C^2-1)\right\}T_{x^4}+O(H)=0\ , \tag{4}$$

as given in Table 9.3.

b) The modified equation for the Crank-Nicolson scheme is applied to the one-dimensional transport equation.

The scheme is

$$\frac{\Delta T_j^{n+1}}{\Delta t}+\{uL_x-\alpha L_{xx}\}\left\{\frac{T_j^n+T_j^{n+1}}{2}\right\}=0\ . \tag{5}$$

Expanding every term of (5) as a Taylor series, we get

$$T_t+uT_x+\frac{\Delta t}{2}T_{tt}+\frac{u\Delta t}{2}T_{tx}-\alpha T_{xx}+\frac{\Delta t^2}{6}T_{t^3}+\frac{u\Delta t^2}{4}T_{t^2x} \\ -\alpha\frac{\Delta t}{2}T_{x^2}+\frac{u\Delta x^2}{6}T_{x^3}+\frac{\Delta t^3}{24}T_{t^4}+\frac{u\Delta t^3}{12}T_{t^3x} \\ -\frac{\alpha\Delta t^2}{4}T_{t^2x^2}+\frac{u\Delta t\ \Delta x^2}{12}T_{tx^3}-\frac{\alpha\Delta x^2}{12}T_{x^4}+O(H)=0\ . \tag{6}$$

From the table, the modified equation is found to be

$$T_t+uT_x-\alpha T_{xx}+\frac{u\Delta x^2}{6}\left(1+\frac{C^2}{2}\right)T_{x^3}-\frac{\alpha\Delta x^2}{12}(1+3C^2)T_{x^4}+O(H) \\ =0\ , \tag{7}$$

as given in Table 9.3.

9.11 a) An analysis of the explicit four-point upwind scheme is carried out.

The scheme is

$$\frac{T_j^{n+1}-T_j^n}{\Delta t}+\{uL_x^{(4)}-\alpha L_{xx}\}T_j^n=0 \tag{1}$$

$$\text{and}\quad L_x^{(4)}=\frac{T_{j+1}-T_{j-1}}{2\Delta x}+\frac{q}{3\Delta x}\left(T_{j-2}-3T_{j-1}+3T_j-T_{j+1}\right). \tag{2}$$

Expanding terms in a Taylor series and simplifying, we have

$$T_t+\frac{\Delta t}{2}T_{tt}+\frac{\Delta t^2}{6}T_{t^3}+\frac{\Delta t^3}{24}T_{t^4}+uT_x+\frac{u\Delta x^2}{3}(0.5-q)T_{x^3} \\ +\frac{uq\ \Delta x^3}{6}T_{x^4}-\alpha T_{xx}-\frac{\alpha\Delta x^2}{12}T_{x^4}+O(H)=0\ . \tag{3}$$

From the tables, we have

$$T_t+uT_x-\alpha T_{xx}+\frac{uC\Delta x}{2}T_{xx}+\left\{\frac{u\Delta x^2}{3}(0.5-q)-\alpha C\Delta x+\right. \\ \left.\frac{C\Delta x}{3}\left(\frac{C^2\ \Delta x^2}{u^2}\right)\right\}T_{x^3}+O(H)=0\ . \tag{4}$$

Comparing (4) with (9.73) (i.e. for the four-point upwind Crank-Nicolson scheme) we find that we now have a first-order dissipation error $(uC\Delta x/2\ T_{xx})$. Hence, highly diffused profiles are to be expected. There does not seem to be any optimal value of q except $q=0.5$ where the dispersion error is reduced somewhat. It can be seen that the dissipation error has arisen from the first-order discretisation of T_t and that there is no compensating term in $L_x^{(4)}T$.

b) von Neumann stability analysis:

Following the procedure outlined before, for the scheme (1), we obtain

$$\begin{aligned} G =& 1 + \frac{u\Delta t}{\Delta x} i \sin\theta + \frac{uq\Delta t}{3\Delta x}\left[\cos 2\theta - 4\cos\theta \right. \\ &\left. + i(-\sin 2\theta + 4\sin\theta)\right] + \frac{2\alpha\Delta t}{\Delta x^2}(\cos^2\theta - 1), \end{aligned} \tag{5}$$

or

$$\begin{aligned} G =& 1+\frac{Cq}{3}(\cos 2\theta - 4\cos\theta) + \frac{2C}{R_{cell}}(\cos\theta - 1) \\ &+ i\left[C\sin\theta + \frac{Cq}{3}(-\sin 2\theta + 4\sin\theta)\right]. \end{aligned} \tag{6}$$

For the conditions of Table 9.6 i.e. $R_{cell} = 1.0$, $q = 0.5$, the scheme is found to be stable for all $C \leq 0.58$.

c) Computational check:

Running TRAN with $R_{cell} = 1.0$, $C = 0.25$, we have the following result for the problem of the propagation of the temperature front:

$x =$	-0.4	-0.2	+0	0.2	0.4	0.6
$T =$	0.9756	0.9401	0.872	0.762	0.609	0.431
$x =$	0.8	1.0	1.2	1.4	1.6	rms error
$T =$	0.258	0.123	0.040	0.005	-0.003	0.0098

The results are identical to those in Table 9.6.

9.12 Modification of TRAN is made to implement the three-level fully implicit scheme, which is given by Table 9.3.

$$\frac{3}{2}\frac{\Delta T_j^{n+1}}{\Delta t} - \frac{1}{2}\frac{\Delta T_j^n}{\Delta t} + \{uL_x - \alpha L_{xx}\}T_j^{n+1} = 0\ . \tag{1}$$

Using $L_xT_j = (T_{j+1} - T_{j-1})/2\Delta x$ and $L_{xx}T_j = (T_{j+1} - 2T_j + T_{j-1})/\Delta x^2$, equation (1) reduces to

$$\begin{aligned} &\left(-\frac{u}{2}\frac{\Delta t}{\Delta x} - \alpha\frac{\Delta t}{\Delta x^2}\right)T_{j-1}^{n+1} \\ &+ \left(\frac{3}{2} + \frac{2\alpha\Delta t}{\Delta x^2}\right)T_j^{n+1} + \left(\frac{u}{2}\frac{\Delta t}{\Delta x} - \frac{\alpha\Delta t}{\Delta x^2}\right)T_{j+1}^{n+1} = 2T_j^n - \frac{1}{2}T_j^{n-1}\ , \end{aligned} \tag{2}$$

which can be written as,

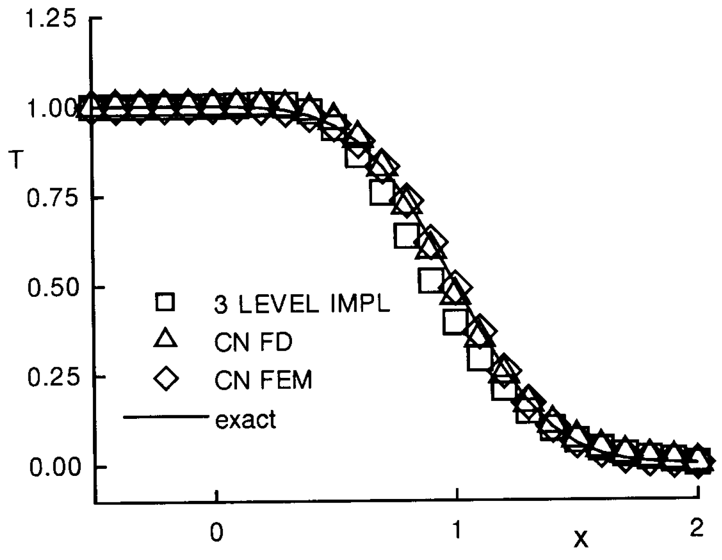

Fig. 1. Temperature profile with $R_{cell} = 2$.

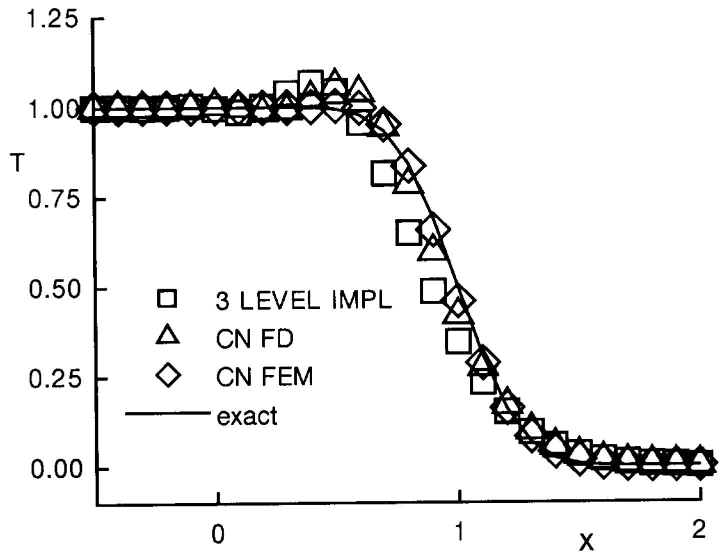

Fig. 2. Temperature profile with $R_{cell} = 5$.

$$\left(-\frac{C}{2}-s\right)T_{j-1}^{n+1}+\left(\frac{3}{2}+2s\right)T_j^{n+1}+\left(\frac{C}{2}-s\right)T_{j+1}^{n+1}=2T_j^n-\frac{1}{2}T_j^{n-1}. \quad (3)$$

Thus the coefficients **AA, BB, CC** and **R** used in TRAN can be modified to suit (3).

Running the modified version of TRAN for the temperature front problem with $JMAX = 41$, NTIM= 10, $C = 1$, $\Delta x = 0.1$, etc, the following solution errors are obtained, for different cell Reynolds number values.

R_{cell}	1	2	5	50	100
3 Level implicit	0.0263	0.0366	0.0588	0.115	0.123
CN-FDM	0.00387	0.00994	0.0286	0.100	0.111
CN-FEM	0.00108	0.00331	0.0110	0.067	0.0783

The temperature profiles computed are shown in the Figs.1 and 2. Again $R_{cell} = 2$ seems to be the limiting cell Reynolds number for an oscillation-free solution.

9.13 An analysis of Crank-Nicolson mass operator formulation is carried out for the two-dimensional transport equation to see if optimal values of δ_x and δ_y can be obtained.

The governing equation is

$$T_t + uT_x + vT_y - \alpha_x\ T_{xx} - \alpha_y\ T_{yy} = 0\ . \quad (1)$$

The corresponding scheme is

$$\begin{aligned}M_x \otimes M_y\left[\frac{T_j^{n+1}-T_j^n}{\Delta t}\right] &= \big[-uM_y \otimes L_x - vM_x \otimes L_y \\ &+ \alpha_x M_y \otimes L_{xx} + \alpha_y\ M_x \otimes L_{yy}\big]\left\{\frac{T_j^{n+1}+T_j^n}{2}\right\}.\end{aligned} \quad (2)$$

Truncation error analysis:

For this analysis it is convenient to expand the terms about $T_j^{n+1/2}$. Expanding the LHS of (2) we have,

$$\begin{aligned}&M_x \otimes M_x\left\{\frac{T_j^{n+1}-T_j^n}{\Delta t}\right\} \\ &= T_t + \frac{\Delta t^2}{6}T_{ttt} + \frac{2\delta_y\ \Delta y^2}{2}T_{tyy} + \frac{2\delta_x\ \Delta x^2}{2}T_{txx} + O(H)\ .\end{aligned} \quad (3)$$

Carrying out the same expansion for the RHS, we have

$$-uM_y \otimes L_xT = -uT_x - \frac{u\Delta x^2}{6}T_{xxx} - \frac{2u\delta_y\ \Delta y^2}{2}T_{xyy} + O(H)\ , \quad (4)$$

$$-vM_x \otimes L_yT = -vT_y - \frac{v\Delta y^2}{6}T_{yyy} - \frac{2v\delta_x\ \Delta x^2}{2}T_{yxx} + O(H)\ , \quad (5)$$

$$\alpha_x M_y \otimes L_{xx}\ T = \alpha_x T_{xx} + \alpha_x\frac{\Delta x^2}{12}T_{x^4} + \alpha_x\delta_y\ \Delta y^2 T_{x^2y^2} + O(H)\ , \quad (6)$$

$$\alpha_y M_x \otimes L_{yy}\ T = \alpha_y\ T_{yy} + \alpha_y\frac{\Delta y^2}{12}T_{y^4} + \alpha_y\delta_x\ \Delta x^2 T_{x^2y^2} + O(H)\ . \quad (7)$$

It remains to replace the time derivatives in (3) by spatial derivatives. For this we have,

$$T_t = -uT_x - vT_y + \alpha_x T_{xx} + \alpha_y\ T_{yy}\ , \quad (8)$$
$$T_{txx} = -uT_{xxx} - vT_{xxy} + \alpha_x\ T_{x^4} + \alpha_y\ T_{x^2y^2}\ , \quad (9)$$
$$T_{tyy} = -uT_{xyy} - vT_{yyy} + \alpha_x\ T_{x^2y^2} + \alpha_y\ T_{y^4}\ , \quad (10)$$

$$\begin{aligned}T_{tt} &= -u\frac{\partial}{\partial x}(T_t) - v\frac{\partial}{\partial y}(T_t) + \alpha_x\frac{\partial^2}{\partial x^2}(T_t) + \alpha_y\frac{\partial^2}{\partial y^2}(T_t) \\ &= u^2T_{xx} + 2uv\ T_{xy} + v^2T_{yy} - 2u\alpha_xT_{xxx} - 2v\alpha_y\ T_{yyy} \\ &- 2v\alpha_x\ T_{xxy} - 2u\alpha_yT_{xyy} + \alpha_x^2T_{x^4} + \alpha_y^2T_{y^4} + 2\alpha_x\alpha_yT_{x^2y^2}\ ,\end{aligned} \quad (11)$$

$$\begin{aligned}T_{ttt} &= -u\frac{\partial}{\partial x}(T_{tt}) - v\frac{\partial}{\partial y}(T_{tt}) + \alpha_x\frac{\partial^2}{\partial x^2}(T_{tt}) + \alpha_y\frac{\partial^2}{\partial y^2}(T_{tt}) \\ &= -u^3\ T_{xxx} - v^3T_{yyy} - 3u^2vT_{xxy} - 3uv^2T_{xyy} \\ &\quad + 3u^2\alpha_xT_{x^4} + 3v^2\alpha_yT_{y^4} + 6uv\alpha_xT_{x^3y} + 6uv\alpha_yT_{xy^3} \\ &\quad\quad + 3(\alpha_yu^2 + \alpha_xv^2)T_{x^2y^2} + O(H)\ .\end{aligned} \quad (12)$$

Consequently, from (3) to (7), we have (after simplification)

$$\begin{aligned}&T_t + uT_x + vT_y - \alpha_xT_{xx} - \alpha_yT_{yy} = \\ &\quad T_{xxx}\left(\frac{u^3\Delta t^2}{6} - \frac{u\Delta x^2}{6} + u\delta_x\ \Delta x^2\right) \\ &\quad + T_{yyy}\left(\frac{v^3\Delta t^2}{6} - \frac{v\Delta y^2}{6} + v\delta_y\Delta y^2\right) \\ &\quad + T_{xxy}\left(u^2v\frac{\Delta t^2}{2}\right) + T_{xyy}\left(uv^2\frac{\Delta t^2}{2}\right) \\ &\quad + T_{x^4}\left(-\frac{u^2\alpha_x\ \Delta t^2}{2} - \delta_x\alpha_x\ \Delta x^2 + \alpha_x\frac{\Delta x^2}{12}\right) \\ &\quad + T_{y^4}\left(-\frac{v^2\alpha_y\ \Delta t^2}{2} - \delta_y\alpha_y\Delta y^2 + \alpha_y\frac{\Delta y^2}{12}\right) \\ &\quad - T_{x^3y}\left(uv\alpha_x\Delta t^2\right) - T_{xy^3}\left(uv\alpha_y\Delta t^2\right) \\ &\quad - T_{x^2y^2}\left(\alpha_yu^2 + \alpha_xv^2\right)\frac{\Delta t^2}{2} + O(H)\ .\end{aligned} \quad (13)$$

The truncation error is of $O(\Delta t^2,\ \Delta x^2,\ \Delta y^2)$.

For the choice $\delta_x = \delta_y = 1/6$, (13) reduces to

$$\begin{aligned} T_t + uT_x + vT_y - \alpha_x T_{xx} - \alpha_y T_{yy} &= \\ T_{xxx}\left(\frac{u^3\Delta t^2}{6}\right) + T_{yyy}\left(\frac{v^3\Delta t^2}{6}\right) &+ T_{xxy}\left(u^2 v\frac{\Delta t^2}{2}\right) \\ + T_{xyy}\left(uv^2\frac{\Delta t^2}{2}\right) &+ O(H)\ . \end{aligned} \qquad (14)$$

It is seen from (13) and (14) that the dispersion errors are minimised when $\delta_x = \delta_y = 1/6$.

von Neumann analysis:

The analysis is carried out on the split formulation given by (9.88) and (9.89) i.e.

$$[M_x + 0.5\Delta t(uL_x - \alpha_x\ L_{xx})]\Delta T^*_{j,k} = \Delta t(RHS)^n, \qquad (15)$$

$$[M_y + 0.5\Delta t(vL_y - \alpha_y\ L_{yy})]\Delta T^{n+1}_{j,k} = \Delta T^*_{j,k}\ . \qquad (16)$$

where the values $\beta = 0.5$ and $\gamma = 0$ have been substituted and

$$RHS = -[M_y \otimes (uL_x - \alpha_x L_{xx}) + M_x \otimes (vL_y - \alpha_y\ L_{yy})]T_{j,k}\ . \qquad (17)$$

On substituting for $\Delta T^*_{j,k}$ from (2), in (1) , we have

$$\begin{aligned} &\Big[M_x + 0.5\Delta t(uL_x - \alpha_x L_{xx})\Big]\Big[M_y + 0.5\Delta t(vL_y - \alpha_y L_{yy})\Big]\Delta T^{n+1}_{j,k} \\ &\quad = -\Delta t\Big[M_y \otimes (uL_x - \alpha_x L_{xx}) + M_x \otimes (vL_y - \alpha_y L_{yy})\Big]T^n_{j,k}\ . \end{aligned} \qquad (18)$$

Introducing the Fourier representation for various operators like,

$$L_{xx} = 2(\cos\theta_x - 1)/\Delta x^2,\ L_x = i\sin\theta_x/\Delta x,\ M_x = 1 + 2\delta_x(\cos\theta_x - 1)\ ,$$

we find the amplification matrix to be given by the following equation:

$$\begin{aligned} &\Big[1 + (1-\cos\theta_x)\{s_x - 2\delta_x\} + i0.5C_x\sin\theta_x\Big] \\ &\quad\Big[1 + (1-\cos\theta_y)\{s_y - 2\delta_y\} + i0.5C_y\sin\theta_y\Big](G-1) \\ &= -\Big[\{1 + 2\delta_y(\cos\theta_y - 1)\}\{s_x(1-\cos\theta_x) + i0.5C_x\sin\theta_x\} \\ &\quad + \{1 + 2\delta_x(\cos\theta_x - 1)\}\{s_y(1-\cos\theta_y) + i0.5C_y\sin\theta_y\}\Big]\ . \end{aligned} \qquad (19)$$

Solving the above equation for G with the aid of a computer program we find that the scheme is unconditionally stable for the parameters chosen (i.e.$\gamma = 0$ and $\beta = 0.5$). This is in agreement with the observation made in the text (see Vol. 1, p.318).

9.14 A comparison is made of the AF-FDM, AF-4PU and AF-FEM schemes for the thermal entry problem.

Running THERM for the conditions indicated the following results are obtained.

AF-FDM, ME=1

	11 x 11 grid			21 x 21 grid		
Parameters	N_{iter}	rms RHS (x10^5)	rms error	N_{iter}	rms RHS (x10^5)	rms error
a) $\gamma = 0, \beta = 0.5$	112	0.86	0.018	177	0.95	0.0068
b) $\gamma = 0, \beta = 1$	32	0.84	0.018	86	0.94	0.0068
c) $\gamma = 0.5, \beta = 1$	20	0.85	0.018	45	0.94	0.0068

AF-4PU

a) $\gamma = 0, \beta = 0.5$	37	0.82	0.036	66	0.71	0.024
b) $\gamma = 0, \beta = 1$	34	0.90	0.036	78	0.99	0.024
c) $\gamma = 0.5, \beta = 1$	23	0.89	0.036	47	0.99	0.024

AF-FEM, ME=3

a) $\gamma = 0, \beta = 0.5$	unst			unst		
b) $\gamma = 0, \beta = 1$	74	0.95	0.0026	190	0.98	0.00083
c) $\gamma = 0.5, \beta = 1$	39	0.96	0.0026	97	0.95	0.00083

In the above table N_{iter} are the number of iterations to converge, rms RHS is the solution error and *unst* denotes that the scheme was unstable. Considering only the number of iterations to converge, we find that AF-FDM with $\gamma = 0.5$ and $\beta = 1.0$ is the fastest. Further, AF-FDM is more accurate than the AF-4PU (see the rms-error). If accuracy is the only consideration, AF-FEM performs the best.

9.15 The computational efficiencies of the various schemes is compared. Defining the computational efficiency as,

$$\eta = 1/\ (\text{rms error})\ \ (\text{No. of iterations to converge})\ .$$

We have for various schemes considered in Problem 9.14,

	AF-FDM		AF- 4PU		AF- FEM	
	11x11	21x21	11x11	21x21	11x11	21x21
$\gamma = 0, \beta = 0.5$	0.5	0.84	0.74	0.64	unstable	unstable
$\gamma = 0, \beta = 1$	1.74	1.72	0.812	0.54	5.12	6.33
$\gamma = 0.5, \beta = 1$	2.77	3.29	1.2	0.9	9.71	12.39

Thus considering the efficiency, the best scheme happens to be the AF-FEM with $\gamma = 0.5$ and $\beta = 1$ for both coarse and fine grids. Table 9.12 shows that this scheme also produces the most accurate solution for the centre-line. If one considers the programming complexity in addition, then the scheme AF-FDM with $\gamma = 0.5$ and $\beta = 1$ is a good choice.

CTFD Solutions Manual: Chapter 10

Nonlinear Convection-Dominated Problems

10.1 A truncation error analysis of (10.20) is carried out as follows.

Equation (10.20) can be written as

$$M_x\left(\frac{\Delta u^{n+1}}{\Delta t}\right) + 0.5\ L_x^{(4)}(u^n u_j^{n+1}) - 0.5\nu L_{xx}(u_j^n + u_j^{n+1}) = 0\ . \tag{1}$$

Writing the term $L_x^{(4)}(u^n u^{n+1})$ as $u^n L_x^{(4)} u^{n+1} + u^n L_x^{(4)} u^n$ and expanding the terms in (1) as a Taylor series about u_j^n gives

$$\begin{aligned} &u_t + u^n u_x + (\Delta t/2)u_{tt} + 0.5\ u^n \Delta t\ u_{tx} - \nu u_{xx} + (\Delta t^2/6)u_{t^3} \\ &\quad + (\Delta t^2/4)u^n u_{ttx} + \left(\delta\Delta x^2 - 0.5\nu\Delta t\right) u_{txx} + u^n\left((1-2q)/6\right)\Delta x^2\ u_{xxx} \\ &\quad + (\Delta t^3/24)u_{t^4} + (u^n\Delta t^3/12)u_{t^3x} + \left(\delta\Delta t(\Delta x^2/2)\right. \\ &\quad \left.- 0.5\nu(\Delta t^2/2)\right)u_{t^2x^2} + 0.5u^n\left((1-2q)/6\right)(\Delta x^2\Delta t/2)u_{tx^3} \\ &\quad + \left((u^n q/6)\Delta x^3 - \nu\Delta x^2/12\right) u_{x^4} = 0\ . \end{aligned} \tag{2}$$

Eliminating the time-derivatives using the governing equations, one has

$$\begin{aligned} &u_t + uu_x - \nu u_{xx} + u^n\left((u^n\Delta t)^2/12 - \delta\Delta x^2 + (1-2q)\Delta x^2/6\right)u_{x^3} \\ &+ O(H) = 0\ . \end{aligned} \tag{3}$$

It is of interest to note that to this order the modified equation is identical to (3).

For the dispersion errors to be minimum we require,

$$u^n\left\{\frac{(u^n\Delta t)^2}{12} - \delta\Delta x^2 + \frac{1-2q}{6}\Delta x^2\right\} = 0\ , \tag{4}$$

which gives (10.26 and 10.27) for $q = 0$ and $\delta = 0$ respectively.

10.2 Modification of BURG to implement the Crank-Nicolson discretisation of (10.1) is carried out as follows,

$$\frac{\Delta u}{\Delta t} + 0.5u^{n+1}\frac{\partial u^{n+1}}{\partial x} + 0.5u^n\frac{\partial u^n}{\partial x} - 0.5\nu\frac{\partial^2 u^{n+1}}{\partial x^2} - 0.5\nu\frac{\partial^2 u^n}{\partial x^2} = 0\ . \tag{1}$$

Linearising $(u^{n+1}\partial u^{n+1}/\partial x)$ as $(u^{n+1}\partial u^{n}/\partial x)+(u^{n}\partial u^{n+1}/\partial x)-(u^{n}\partial u^{n}/\partial x)$, and simplifying, leads to

$$u^{n+1}+0.5\ \Delta t\ u^{n}\frac{\partial u^{n+1}}{\partial x}+0.5\ \Delta t\ u^{n+1}\frac{\partial u^{n}}{\partial x}-0.5\nu\frac{\partial^2 u^{n+1}}{\partial x^2}= u^{n}+0.5\nu\frac{\partial^2 u^{n}}{\partial x^2}\ . \qquad (2)$$

Now, discretising $\partial u/\partial x$ and $\partial^2 u/\partial x^2$ and simplifying, we have,

$$(-0.25Cu^n-0.5\ s)u_{j-1}^{n+1}+(1+0.25C(u_{j+1}^n-u_{j-1}^n)+s)u_j^{n+1} + (0.25Cu^n-0.5\ s)u_{j+1}^{n+1} = (0.5s)u_{j-1}^n+(1-s)u_j^n+(0.5s)u_{j+1}^n\ , \qquad (3)$$

where $C=\Delta t/\Delta x$ and $s=\nu\Delta t/\Delta x^2$. Equation (3) is the tridiagonal equation to be solved.

Modification to the program BURG to implement (3) consists of replacing the lines 121-129 by FORTRAN statements equivalent to (3).

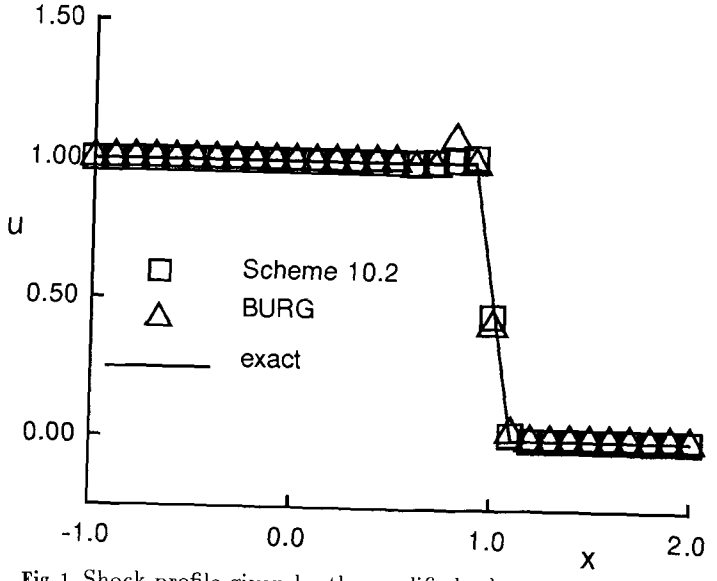

Fig. 1. Shock profile given by the modified scheme.

On running the modified program for the conditions associated with Table 10.5, we get the shock profile shown in Fig. 1 at t=2. The various conditions imposed were $R_{cell}=100$, $\alpha=0.001$, $\Delta x=\Delta t=0.1$, $s=0.1$. Since the cell Reynolds number is high, the scheme needs extra artificial dissipation to produce a good shock profile. The modified scheme, with $SA=0.25$ gave a 25% overshoot near the shock and an rms error of 0.0809. But with $SA=0.5$, the overshoot is negligible (as seen in Fig. 1) and the rms error is 0.0098. In comparison, the basic scheme BURG gives rms errors of 0.0204 and 0.0322 with $SA=0.25$ (Fig. 1) and 0.5 respectively.

When the computations are carried out for larger number of time steps, it is found that the shock travels right to the downstream boundary. With the Dirichlet type boundary conditions on the downstream boundary large overshoots are observed. A gradient boundary condition may give a different result. For instance, if the condition, $(du/dx)\ =\ 0$ is imposed it will result in the shock being swept out of the domain giving a uniform flow and a very low rms error.

10.3 Program BURG is readily modified to implement the optimal values of δ and q, i.e.,

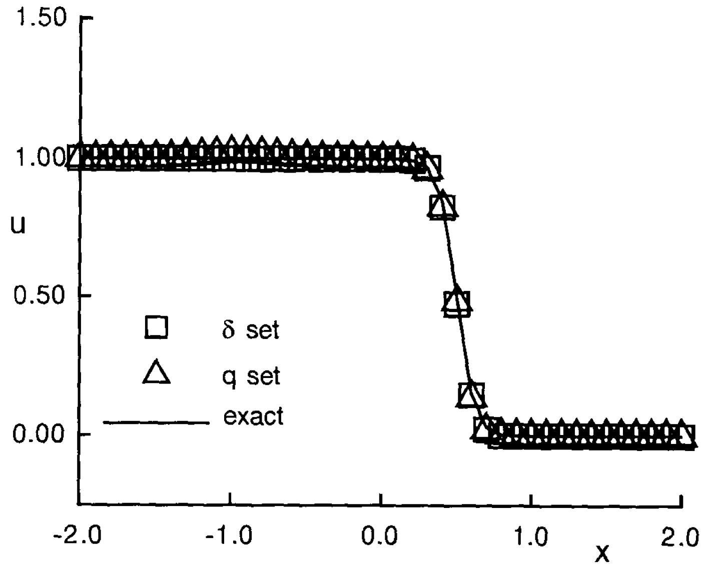

Fig. 1. Shock profile obtained with optimum values of δ and q, $R_{cell}=3.33$.

$$\delta_{opt}=\frac{1}{6}+\frac{(u_j\Delta t/\Delta x)^2}{12} \qquad (1)$$

$$\text{and}\quad q_{opt}=0.5+\frac{(u_j\Delta t/\Delta x)^2}{4}\ . \qquad (2)$$

The changes in the code consist of:

1. Introducing statements for (1) and (2) after line 118.

2. Then calculating the terms depending on δ and q. This requires lines 49-62 to be moved and placed after the statements for (1) and (2).

The program is run for $ME=4$, $R_{cell}=3.33$. After NTIM= 1, and all the other conditions as in Table 10.4 the computed results are as follows: rms error

$= 0.00553$ for δ given by (1) and rms error $= 0.00826$ for q given by (2). These results are plotted in Fig.1.

It is observed that the shock profiles for the two cases are in very good agreement with the exact solution and the computed rms error is also small. Thus, for the cell Reynolds number considered ,i.e. 3.33, the optimum values of δ and q are seen to be effective.

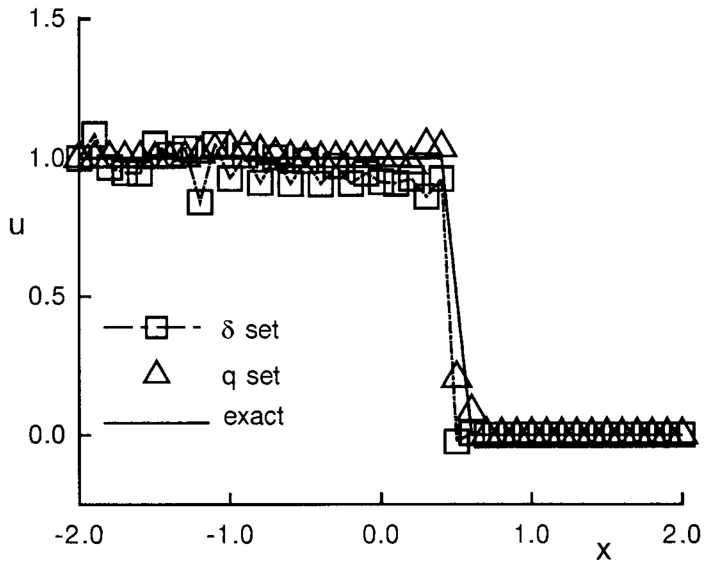

Fig. 2. Shock profile obtained with optimum values of δ and q, $R_{cell} = 100.0$.

Running the same program with the optimum value of δ or q at $R_{cell} = 100$ leads to a different conclusion (see Fig.2). Then the rms error for the δ-adjusted and the q-adjusted schemes are 0.18065 and 0.096319. The former scheme is seen to give a very oscillatory solution indicating that the optimum value of δ is effective only for low cell Reynolds numbers.

10.4 A general three-level scheme to compute steady-state shock solutions can be developed as follows. The governing equation is

$$\frac{\partial u}{\partial t} + \frac{\partial F}{\partial x} - \nu \frac{\partial^2 u}{\partial x^2} = 0 \ , \tag{1}$$

with $F = 0.5(u^2 - u)$.

Introducing a three-level scheme into (1), we have

$$\frac{(1+\gamma)\Delta u^{n+1}}{\Delta t} - \gamma \frac{\Delta u^n}{\Delta t} + (1-\beta)\frac{\partial F^n}{\partial x} + \beta \frac{\partial F^{n+1}}{\partial x} - (1-\beta)\nu \frac{\partial^2 u^n}{\partial x^2} - \beta\nu \frac{\partial^2 u^{n+1}}{\partial x^2} = 0 \ . \tag{3}$$

Linearising F^{n+1} we get, $F^{n+1} = (u^n - 0.5)u^{n+1} - 0.5(u^n)^2$. (4)

Substituting (4) in (3) and rearranging, one gets

$$\begin{aligned} &\{-\beta C(u_{j-1}^n - 0.5) - \beta s\}u_{j-1}^{n+1} + \{(1+\gamma) + 2\beta s\}u_j^{n+1} \\ &+ \{\beta C(u_{j+1}^n - 0.5) - \beta s\}u_{j+1}^{n+1} = (1+2\gamma)u_j^n \\ &- 0.5(1-\beta)C(F_{j+1}^n - F_{j-1}^n) + (1-\beta)s(u_{j-1}^n - 2u_j^n + u_{j+1}^n) \\ &+ 0.25\beta C\{(u_{j+1}^n)^2 - (u_{j-1}^n)^2\} - \gamma u_j^{n-1} \ . \end{aligned} \tag{5}$$

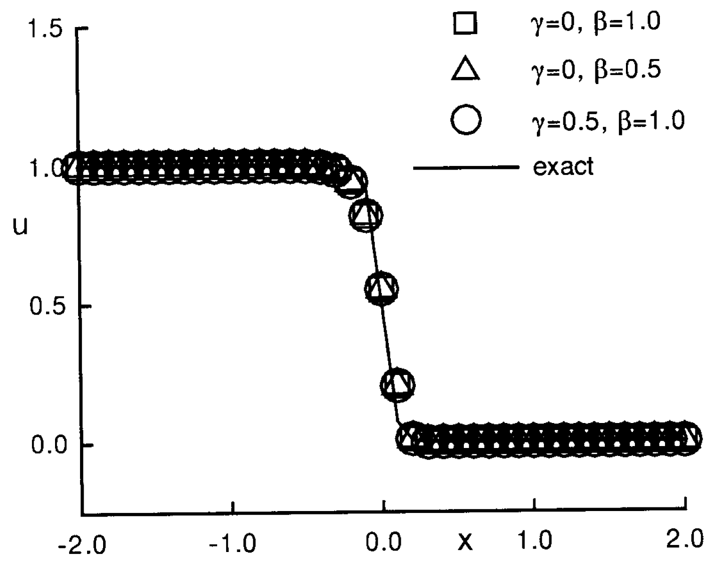

Fig. 1. Shock profile obtained with the modified scheme.

Program BURG is modified to implement (5) and the structure of the program is similar to that of TRAN in Chapter 9. It may also be pointed out that now the exact solution is given by (10.36).

The modified program is run for the conditions corresponding to Table 10.9 and the computed rms errors are as follows. The solution is assumed to have converged when the rms change is less than 1×10^{-6}. In the table N_i denotes the number of iterations to converge and the following parameters are used: $\Delta x = 0.1$, $\Delta t = 0.1$, $C = 1$, $s = 0.2$, $\alpha = 0.02$ $R_{cell} = 5$, $SA = 0.25$.

Scheme	N_i	rms error
$\gamma = 0,\ \beta = 1.0$	46	0.02927
$\gamma = 0,\ \beta = 0.5$	41	0.02927
$\gamma = 0.5, \beta = 1.0$	42	0.02927

The different schemes used take about the same number of iterations to converge, except when $\gamma = 0.5$, $\beta = 1$. The computed steady-state shock profiles (Fig. 1) are identical for the three schemes since $u_j^n = u_j^{n+1}$. Further, the grid points within the shock profile contribute most to the solution error.

10.5 To code the nonuniform discretisation (10.30-10.32) the scheme developed for Problem 9.7 serves as a good starting point. In the present case it is to be noted that R_{cell} is a function of u and hence varies from point to point. The relevant part of the listing is indicated below. Running the modified program leads to the results shown in Table 10.6 and 10.7 being obtained.

[NOTE: The expression $(r_x - 1)/r_x$ in (10.33) should be corrected as $(r_x^2 - 1)/r_x$].

Modifications to TRAN for Problem 10.5

1. The READ statements are changed to read only ME, JMAX, ALPH, UBALPH (i.e. u/α), RX.
2. Lines 18 to 29 are replaced by,

```
      RXX=1./RX
      JMAP = JMAX - 1
      JMAF = JMAX - 2
      XINT=0.0
      XEND=1.0
      DELX=(XEND-XINT)/(JMAX-1)
C
C     CALCULATE X
C
      X(JMAX)=XEND
      DX=0.03
      DO 30  J=JMAP,1,-1
      IF(J.EQ.JMAP) THEN
      DXX=DX
      ELSE
      DXX=DXX*RXX
      ENDIF
   30 X(J)=X(J+1)-DXX
C
C     CALCULATION OF RCELL
C
      U=UBALPH*ALPH
      RCELL = U*DELX/ALPH
```

3. Lines 54 to 147 are replaced by:

```
C
      DO 9 J = 1,JMAX
    9 T(J)=0.0
C
C     SET BOUNDARY CONDITIONS
C
      T(1) = 0.
      T(JMAX) = 1.
C     SET UP THE TRIDIAGONAL SYSTEM OF EQUATIONS
C
      A(1,1) = 0.
      A(2,1) = 0.
      A(3,1) = BD
      A(4,1) = CD
      A(5,1) = 0.
C
      DO 14 J = 2,JMAP
      DELX=X(J+1)-X(J)
      RCELL = U*DELX/ALPH
C
      AD = -(1.+0.5*RCELL*RX)
      CD=0.5 * RXX * RCELL - RXX
      BD=0.5*RCELL*RXX*(RX-1.)+1.+RXX
C
      JM = J - 1
      A(1,JM) = 0.
      A(2,JM) = AD
      A(3,JM) = BD
      A(4,JM) = CD
      A(5,JM) = 0.
      D(JM) = 0.
   14 CONTINUE
      D(1) = -AD*T(1)
      D(JMAF) =  - CD*T(JMAX)
      CALL BANFAC(A,JMAF,1)
C
      CALL BANSOL(D,DUM,A,JMAF,1)
C
      DO 15 J = 2,JMAP
   15 T(J) = DUM(J-1)
```

```
C
C     OBTAIN EXACT SOLUTION AND COMPARE
C
      DENM=EXP(UBALPH)-1.0
      DO 118 J=1,JMAX
  118 TE(J)=(EXP(UBALPH*X(J))-1.0)/DENM
```

10.6 The scheme incorporating a four-point upwind discretisation on a nonuniform grid for the modified Burgers' equation is derived as follows. Let

$$\frac{\partial F}{\partial x} = \frac{(F_{j+1} - F_{j-1})}{(1+r)\Delta x} + aF_{j-2} + bF_{j-1} + cF_j + dF_{j+1} \ , \tag{1}$$

where $\Delta x = x_j - x_{j-1}$, $r_x = (x_{j+1} - x_j)/\Delta x$ and $r_1 = (x_{j-1} - x_{j-2})/\Delta x$. Expanding every term in (1) as a Taylor series about T_j, we have

$$a + b + c + d = 0 \ , \tag{2}$$

$$-a(1+r_1) - b + dr_x = 0 \ , \tag{3}$$

$$a(1+r_1)^2 + b + dr_x^2 + (r_x - 1)/\Delta x = 0 \ . \tag{4}$$

Solving these equations we have,

$$a = -\Big\{d\{r_x(1+r_x)\} + (r_x - 1)/\Delta x\Big\}\Big/\Big\{r_1(1+r_1)\Big\},$$

$$b = \Big\{(r_x - 1)/\Delta x + dr_x(1 + r_x + r_1)\Big\}\Big/ r_1,$$

$$c = \Big\{(1 - r_x)/\Delta x - d(1+r_x)(1 + r_x + r_1)\Big\}\Big/(1+r_1) \ ,$$

where d has been treated as the 'free parameter'. Incorporating the above discretisation for $\partial F/\partial x$ into (10.39) the following quadri-diagonal equation is obtained,

$$\begin{aligned} &\Big(a(u_{j-2} - 0.5)\Delta t\Big)\Delta u_{j-2} \\ &+ \Big(b - \frac{1}{(r_x+1)\Delta x}\Delta t(u_{j-1} - 0.5) - \frac{2s}{(r_x+1)}\Big)\Delta u_{j-1} \\ &\Big(1 + \Delta t(u_j - 0.5)c + \frac{2s}{r_x}\Big)\Delta u_j \\ &\Big(\Delta t(u_{j+1} - 0.5)\big[d + \frac{1}{(1+r_x)\Delta x} - \frac{2s}{r_x(1+r_x)}\big]\Big)\Delta u_{j+1} \\ &= RHS \ , \end{aligned} \tag{5}$$

where RHS is given by

$$\begin{aligned} &- \Delta t\Big\{\frac{(F_{j+1} - F_{j-1})}{(1+r_x)\Delta x} + aF_{j-2} + bF_{j-1} + cF_j + dF_{j+1}\Big\} \\ &+ 2s\Big\{\frac{u_{j-1}}{(r_x+1)} - \frac{u_j}{r_x} + \frac{u_{j+1}}{r_x(r_x+1)}\Big\} \ . \end{aligned} \tag{6}$$

The above equation is solved using subroutines BANFAC and BANSOL as before. The listing of the program developed to do this is given below.

Modifications to BURG for Problem 10.6.

1. Extra arrays UP, UR and F, each with the same dimension as U are introduced and the variables RX, DXMIN, GAM, BET and DFREE are read.

2. Lines 19 to 21 are deleted and lines 47 to 67 are replaced by:

```
      JHLF=(JMAX+1)/2
      X(JHLF)=0.0
      DO 33 J=JHLF,JMAP
      DX=DX*RX
      IF(J.EQ.JHLF) DX=DXMIN
      X(J+1)=X(J)+DX
   33 X(JMAX-J)=-X(J+1)
C
C     INITIALISE U AND EVALUATE UEX
C
      CALL EXSH(JMAX,X,U,UE,DX,XMAX,ALPH,TIMAX)
```

3. Lines 70 and 71, 86 to 114 are deleted. The following lines are inserted after 84.

```
      DO 27 N = 1,NTIM
      DO 271 J=1,JMAX
  271 F(J)=0.5*U(J)*(U(J)-1.0)
```

4. Lines 116 to 172 are replaced by:

```
C
C     TRIDIAGONAL SYSTEM FOR IMPLICIT SCHEMES
C
   21 IF(MQ .EQ. 1)DIM = U(1)
      DO 22 J = 2,JMAP
      DX=X(J)-X(J-1)
      RX=(X(J+1)-X(J))/DX
      RX1=(X(J-1)-X(J-2))/DX
      S=ALPH*DT/DX/DX
      RXP=RX + 1.
      RXM=RX - 1.
      RXMD=RXM/DX
      RXPD=1./(RXP*DX)
      RXMP=(RX -1.)/(RX+1.)
C
```

```
      AA = -(RXMD + RX*RX*DFREE + RX*DFREE)/(RX1*(RX1+1))
      BB= (RXMD + RX*RX*DFREE +(RX1+1.)*RX*DFREE)/RX1
      CC=-DFREE - AA - BB
C
      JM = J - 1
C
      FLUX = (F(J+1)-F(J-1)) * RXPD
      IF(J.NE.2)
     1 FLUX = FLUX + AA*F(J-2) + BB*F(J-1)
     1 + CC*F(J) +DFREE*F(J+1)
       SEC=2.*S*(U(JM)/RXP - U(J)/RX + U(J+1)/RX/RXP)
C
      DIM = U(1)
      IF(JM .NE. 1)DIM = U(JM-1)
      A(1,JM) = AA*(DIM   - 0.5)*DT
      A(2,JM) = (BB - RXPD)*DT*(U(J-1)-0.5) - 2.*S/RXP
      A(3,JM) = 1. + DT*(U(J)-0.5)*CC + 2.*S/RX
      A(4,JM) = DT*(U(J+1)-0.5)*(DFREE+RXPD) - 2.*S/RX/RXP
      IF(ME .NE. 5 )GOTO 22
      A(2,JM) = A(2,JM) - 0.5*SA*U(JM)
      A(3,JM) = A(3,JM) + SA*U(J)
      A(4,JM) = A(4,JM) - 0.5*SA*U(J+1)
   22 R(JM) =  - DT*FLUX + SEC
C
C     REDUCE A TO TRIDIAGONAL FORM
C
      DO 23 JM = 3,JMAF
      JMM = JM - 1
      DUM = A(1,JM)/A(2,JMM)
      A(2,JM) = A(2,JM) - A(3,JMM)*DUM
      A(3,JM) = A(3,JM) - A(4,JMM)*DUM
      A(1,JM) = 0.
      R(JM) = R(JM) - R(JMM)*DUM
   23 CONTINUE
      A(1,1) = 0.
      A(1,2) = 0.
   24 A(2,1) = 0.
      A(4,JMAF) = 0.
C
      CALL BANFAC(A,JMAF,1)
      CALL BANSOL(R,UD,A,JMAF,1)
C
   25 DO 26 J = 2,JMAP
      IF(ME .GT. 3)UR(J) = U(J)+UD(J-1)
   26 CONTINUE
      SUM=0.0
```

```
      DO 37 J=2,JMAP
      SUM=SUM + (UR(J)-U(J))**2
   37 U(J)=UR(J)
      RMS=SQRT(SUM/(AMP-1.))
      WRITE(*,*)N,RMS
      IF(RMS .LT. 1E-06) GO TO 227
   27 CONTINUE
  227 CONTINUE
C
      WRITE(6,28)TIMAX
   28 FORMAT(' FINAL SOLUTION,    TIM =',F5.3)
      WRITE(6,16)(X(J),J=1,JMAX)
      WRITE(6,17)(U(J),J=1,JMAX)
C     WRITE(6,29)(UE(J),J=1,JMAX)
   29 FORMAT('  UE=',12F6.3)
C   COMPUTE THE EXACT SOLUTION AND COMPARE.
      SUM = 0.
      DO 30 J = 2,JMAP
      ARGG=0.25*X(J)/ALPH
      EXPA=EXP(ARGG)
      EXPP=1/EXPA
      TANHH=(EXPA - EXPP)/(EXPA + EXPP)
      UE(J)=0.5*(1. - TANHH)
   30 SUM = SUM + (U(J) - UE(J))**2
      RMS = SQRT(SUM/(AMP-1.))
      WRITE(6,31)RMS
   31 FORMAT(' RMS ERR=',E12.5)
      DO 555 J=1,JMAX
555   WRITE(6,666)J,X(J),U(J),UE(J)
666   FORMAT(2X,I4,2X,E12.5,2X,E12.5,2X,E12.5)
      STOP
      END
```

The program was run for various values of r_x and the free parameter d. The rms solution errors are shown below.

$r_x \setminus$ d=	-1.0	-0.1	0.0	0.1
1.0	0.02258	0.00616	0.00552	0.02006

$r_x \setminus$ d=	-0.68	-0.5	0.0	0.1
1.1	0.01032	0.21968	0.42782	0.42962

$r_x \setminus$ d=	-1.0	-0.5	0.0	0.5
1.2	0.60309	0.010528	0.51973	0.48565

$r_x \setminus$ d=	-1.0	-0.5	0.0	0.5
1.3	0.62795	0.14526	0.59168	0.54862

Thus , we see that the scheme behaviour is very sensitive to the choice of r_x and the free parameter d. In the computations various values of the parameter d were tested. In the above table results for values of d producing the minimum error for the particular value of r_x and those for values of d where instability is observed, have been given.

10.7 The modifications required in program SHOCK to implement the explicit four-point upwind scheme are shown in the listing.

The modified program is run for conditions of Fig. 10.9, i.e. shock pressure ratio = 1.5, N_x = 101 etc. The artificial viscosity parameter, ENL, was set to be 0.25. A value q = 0.5 is seen to produce a better shock profile than any lower value (say 0.2). Considering the higher values, q=1.0 produces an oscillatory solution whereas q=1.5 makes the scheme unstable for these conditions.

Modifications to SHOCK for Problem 10.7

The parameter QUP is read in and lines 101 to 113 are deleted. Lines 94 to 96 are replaced by,

```
C
      DO 13 K = 1,3
      Q(J,K) = Q(J,K) - 0.5*DTR*(F(J+1,K) - F(J-1,K))
      IF(J-2 .GT. 0) THEN
      Q(J,K)=Q(J,K)-(QUP*DTR/3.)*(F(J-2,K)-3.*F(J-1,K)
     1 +3.*F(J,K)-F(J+1,K))
      ENDIF
```

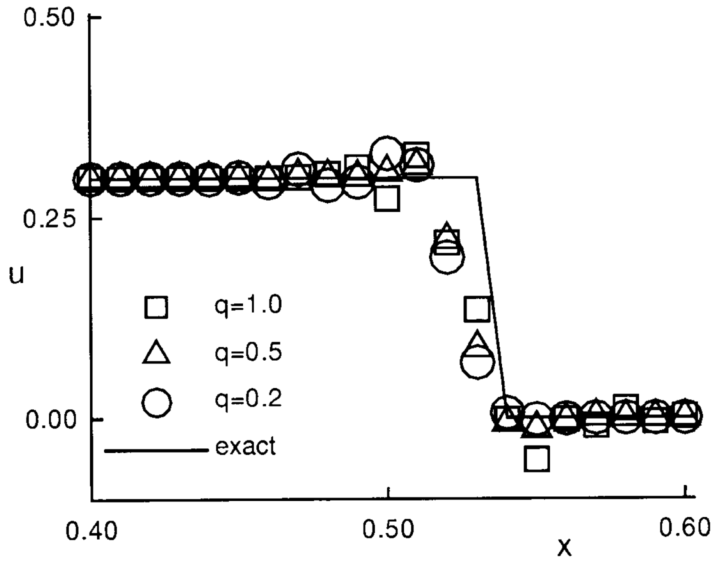

Fig. 1. Shock profile given by the four-point upwind scheme.

10.8 Introducing the mass operator into (10.44), we obtain

$$M_x \otimes \Delta q^{n+1} = -0.25\frac{\Delta t}{\Delta x}\left[(F^n_{j+1} - F^n_{j-1}) + (F^{n+1}_{j+1} - F^{n+1}_{j-1})\right] . \qquad (1)$$

Carrying out the linearisation as in (10.45), we have

$$\begin{aligned}&\left(\delta - 0.25\frac{\Delta t}{\Delta x}A_{j-1}\right)\Delta q^{n+1}_{j-1} + (1-2\delta)\; I\; \Delta q^{n+1}_j + \\ &(\delta + 0.25\frac{\Delta t}{\Delta x}A_{j+1})\Delta q^{n+1}_{j+1} \\ &= RHS\ of\ (10.46) .\end{aligned} \qquad (2)$$

The Jacobian $\underline{\mathbf{A}}$ is given by,

$$\underline{\mathbf{A}} = \begin{bmatrix} 0 & 1 & 0 \\ 0.5(\gamma-3)u^2 & (3-\gamma)u & \gamma-1 \\ u(-\gamma E/\,\rho + (\gamma-1)u^2) & \gamma E/\,\rho - 0.5(\gamma-1)3u^2 & \gamma u \end{bmatrix} . \qquad (3)$$

Now, (2) is block tridiagonal and the Thomas algorithm (Sect. 6.2.5) is used to solve it. The program SHOCK is modified accordingly and the listing of the modified version is given below. This version of SHOCK calls the subroutine BLTRI which solves the block tridiagonal equations. The listing of this subroutine is given under Problem 6.6.

The program is run to compute the propagation of the shock of strength 1.5 for different values of δ. It was found that without artificial viscosity the scheme gives post-shock oscillations and hence the value of ENL is set equal to 0.5 in all calculations.

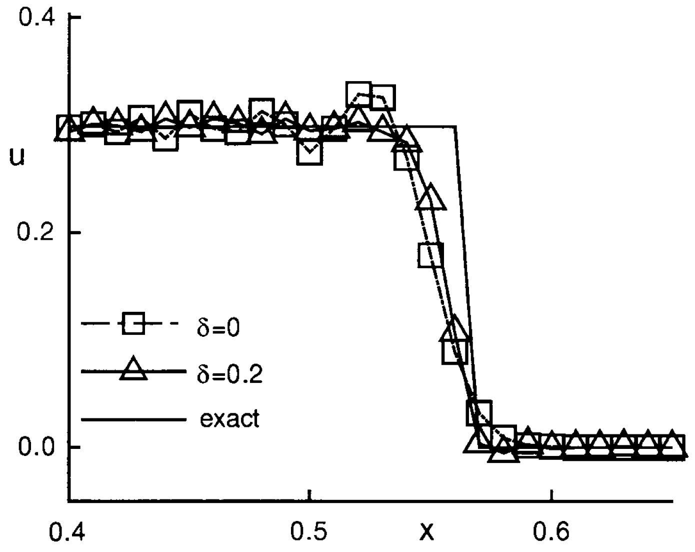

Fig. 1. Shock profile given by the modified SHOCK program.

It is observed that with $\delta = 0$, and 0.1 there are still oscillations left undamped (typically 10%). For $\delta = .16, .2$ the profile is free of oscillations and the shock profile is spread around 3 or 4 mesh widths.

Modifications to SHOCK for Problem 10.8

1. The following line is appended to the DIMENSION statement,
```
     1 ,AA(101,3,3),RHS(101,3),A(101,3,3),B(101,3,3),DQ(101,3)
     1 ,C(101,3,3)
```
2. Lines 85 to 89 are deleted and lines 93 to 114 are replaced by,

```
C    CALCULATE THE JACOBIANS.
C
      DO 18 J = 1,NX
      EE=GAM*Q(J,3)/RH(J)
      USQ=U(J)*U(J)
      AA(J,1,1)=0.
      AA(J,1,2)=1.
      AA(J,1,3)=0.
```

```
      AA(J,2,1)=0.5*(GAM-3.)*USQ
      AA(J,2,2)=(3.-GAM)*U(J)
      AA(J,2,3)=GMM
      AA(J,3,1)=U(J)*(-EE+ GMM*USQ)
      AA(J,3,2)= EE-0.5*GMM*3.*USQ
      AA(J,3,3)=GAM*U(J)
      DO 18 K=1,3
      DO 18 L=1,3
18    AA(J,K,L)=0.25*DTR*AA(J,K,L)
C
C    CALCULATE THE LEFT HAND SIDE.
C
      NXMM=NXM - 1
C
      DO 19 J = 2,NXM
      JM = J-1
      DO 19 K=1,3
      RHS(JM,K)=-0.5*DTR*(F(J+1,K)-F(J-1,K))
      DO 19 L=1,3
      A(JM,K,L)=   - AA(J-1,K,L)
      B(JM,K,L)=0.0
      C(JM,K,L)=    + AA(J+1,K,L)
      IF(K.EQ.L) THEN
      B(JM,K,L)=1.0 -2.*EM
      A(JM,K,L)=EM + A(JM,K,L)
      C(JM,K,L)=EM + C(JM,K,L)
      ENDIF
  19  CONTINUE
      DO 119 K=1,3
      DO 119 L=1,3
      A(1,K,L)=0.0
 119  C(JM,K,L)=0.0
C
C    CALL BLTRI TO SOLVE THE BLOCK TRIDIAGONAL EQUATIONS.
C
      CALL BLTRI(NXMM,A,B,C,RHS,DQ)
      DO 120 J=2,NXM
      DO 120 K=1,3
 120  Q(J,K)=Q(J,K) + DQ(J-1,K)
```

10.9 The conventional finite-element method applied to (10.1) differs from the group finite-element method only in the treatment of the term $u\partial u/\partial x$. Using (10.51) along with Crank-Nicolson time differencing we have,

$$M_x \otimes \frac{\Delta u^{n+1}}{\Delta t} = -0.5\left(\frac{u_{j+1}+u_j+u_{j-1}}{3}\right)^{n+1}\left(\frac{u_{j+1}-u_{j-1}}{2\Delta x}\right)^{n+1} - 0.5\left(\frac{u_{j+1}+u_j+u_{j-1}}{3}\right)^{n}\left(\frac{u_{j+1}-u_{j-1}}{2\Delta x}\right)^{n} + \frac{0.5\nu}{\Delta x^2}(u_{j+1}-2u_j+u_{j-1})^n + \frac{0.5\nu}{\Delta x^2}(u_{j+1}-2u_j+u_{j-1})^{n+1} \quad (1)$$

Linearising terms like $A^{n+1}F^{n+1}$ as $A^nF^{n+1} + A^{n+1}F^n - A^nF^n$ and simplifying, we obtain the tridiagonal equation:

$$u_{j-1}^{n+1}\left(\delta + \frac{0.5C}{6}(U_{diff}^n - U_{avg}^n) - 0.5s\right) + u_j^{n+1}\left(1 - 2\delta + \frac{0.5C}{6}U_{diff}^n + s\right) u_{j+1}^{n+1}\left(\delta + \frac{0.5C}{6}(U_{avg}^n + U_{diff}^n) - 0.5s\right) = \delta u_{j-1}^n + (1-2\delta)u_j^n + \delta u_{j+1}^n + 0.5C(u_{j+1}-2u_j+u_{j-1})^n, \quad (2)$$

where $U_{avg} = (u_{j+1}+u_j+u_{j-1})/3$ and $U_{diff} = (u_{j+1}-u_{j-1})$.

Program BURG is modified to implement the scheme given by (2). The parts of the program that require to be changed are the calculation of the RHS and the left hand side. The actual FORTRAN statements used and other changes are shown in the following listing.

The shock profile shown in Fig. 1 was calculated by the modified scheme incorporating the conventional finite element method for conditions of Table 10.5. The artificial viscosity parameter SA was set equal to 0.25. The rms error computed is 0.02121. It is observed that the shock profile is sharp, but with a larger overshoot and undershoot than for either $CN-MO$ or $CN-FEM$.

Modifications to BURG for Problem 10.9

1. After line 29 the following statement is added,

```
C56=0.5*C/6.
```

2. Lines 118 to 132 are replaced by,

```
DO 22 J = 2,JMAP
UAVG = U(J-1) + U(J) + U(J+1)
UDIFF=U(J+1) - U(J-1)
JM = J - 1
IF(JM .NE. 1)DIM = U(JM-1)
A(1,JM) = 0.5*QQ*DIM
A(2,JM) = EM - 0.5*S - C56*UAVG + C56*UDIFF
A(3,JM) = 1.0 - 2.0*EM + S + C56*UDIFF
A(4,JM) = EM - 0.5*S  + C56*UAVG + C56*UDIFF
```

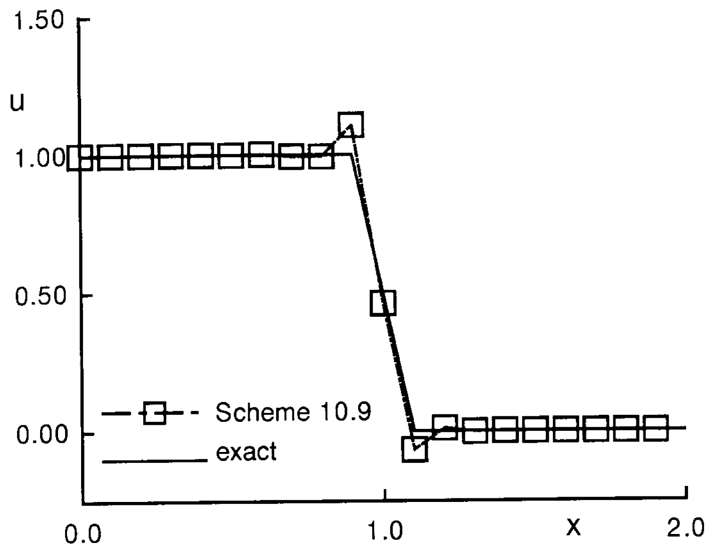

Fig. 1. Shock profile given by the conventional finite element method.

```
   IF(ME .NE. 5 )GOTO 22
   A(2,JM) = A(2,JM) - 0.5*SA*U(JM)
   A(3,JM) = A(3,JM) + SA*U(J)
   A(4,JM) = A(4,JM) - 0.5*SA*U(J+1)
22 R(JM) = EM*(U(J-1)+U(J+1)) + (1.-2.*EM)*U(J)
 1        +0.5*S*(U(J+1)-2.*U(J)+U(J-1))
   R(1) = R(1) - A(2,1)*U(1)
```

10.10 The connectivity (CON) and residual operation count ($ROPC$) for the group ($FEM(G)$) and conventional finite element ($FEM(C)$) methods applied to the two-dimensional Burgers' equations with quadratic interpolation are shown in the table.

	$FEM(C)$		$FEM(G)$	
node	CON	ROPC	CON	ROPC
(j,k)	289	4924	25	700
$(j-1,k)$	99	1764	15	420
$(j-1,k-1)$	99	1404	9	252

The connectivity depends on how many elements are connected to the Galerkin node. In addition on a regular grid or when evaluating $u\partial u/\partial x$ the connectivity will be slightly less. However it is clear that the connectivity is substantially reduced for the group finite element method and there is a corresponding reduction in the number of operations to evaluate the residuals.

10.11 The conventional finite element method is applied to (10.57) and (10.58) for the linear finite elements (A,B,C,D) in Fig.1. We find that the linear terms behave like the corresponding terms in the group finite element method, but the convective terms need special treatment. The discretisation for typical convective terms is as follows:

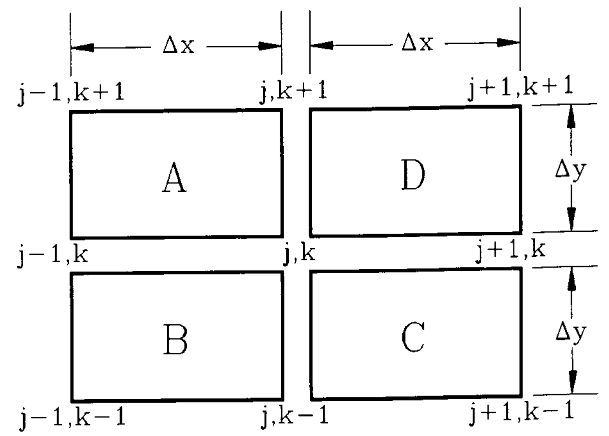

Fig.1. Linear finite elements.

$$\begin{aligned} u\frac{\partial v}{\partial x} = & \frac{M_{y3}}{2\Delta x}\left\{u^*_{j+1,k+1}v_{j+1,k+1} - u^*_{j-1,k+1}v_{j-1,k+1} - \Delta u^*_{j,k+1}v_{j,k+1}\right\} \\ & + \frac{M_{y2}}{2\Delta x}\left\{u^*_{j+1,k}v_{j+1,k} - u^*_{j-1,k}v_{j-1,k} - \Delta u^*_{j,k}v_{j,k}\right\} \\ & + \frac{M_{y1}}{2\Delta x}\left\{u^*_{j+1,k-1}v_{j+1,k-1} - u^*_{j-1,k-1}v_{j-1,k-1} - \Delta u^*_{j,k-1}v_{j,k-1}\right\} , \end{aligned} \qquad (1)$$

where

$$M_y = \{\frac{1}{6}, \frac{2}{3}, \frac{1}{6}\}^T ,$$

$$u^*_{j+1,k+1} = \left\{\frac{u_{j,k}}{3} + \frac{u_{j+1,k}}{6} + \frac{u_{j+1,k+1}}{6} + \frac{u_{j,k+1}}{3}\right\} ,$$

$$u^*_{j-1,k+1} = \left\{\frac{u_{j,k}}{3} + \frac{u_{j-1,k}}{6} + \frac{u_{j-1,k+1}}{6} + \frac{u_{j,k+1}}{3}\right\} ,$$

$$u^*_{j+1,k} = \left\{\frac{u_{j,k}}{2} + \frac{u_{j+1,k}}{4} + \frac{u_{j,k+1}}{12} + \frac{u_{j,k-1}}{12} + \frac{u_{j+1,k-1}}{24} + \frac{u_{j+1,k+1}}{24}\right\} ,$$

$$u^*_{j-1,k} = \left\{\frac{u_{j,k}}{2} + \frac{u_{j-1,k}}{4} + \frac{u_{j,k+1}}{12} + \frac{u_{j,k-1}}{12} + \frac{u_{j-1,k-1}}{24} + \frac{u_{j-1,k+1}}{24}\right\} ,$$

$$u^*_{j+1,k-1} = \left\{\frac{u_{j,k}}{3} + \frac{u_{j+1,k}}{6} + \frac{u_{j+1,k-1}}{6} + \frac{u_{j,k-1}}{3}\right\} ,$$

$$u^*_{j-1,k-1} = \left\{\frac{u_{j,k}}{3} + \frac{u_{j-1,k}}{6} + \frac{u_{j-1,k-1}}{6} + \frac{u_{j,k-1}}{3}\right\} ,$$

$$\Delta u^*_{j,k+1} = \frac{1}{6}\left\{u_{j+1,k} + u_{j+1,k+1} - u_{j-1,k} - u_{j-1,k+1}\right\}$$

$$\begin{aligned} \Delta u^*_{j,k} = & \frac{1}{4}\left[\left\{\frac{1}{6}u_{j+1,k-1} + u_{j+1,k} + \frac{1}{6}u_{j+1,k+1}\right\}\right. \\ & \left. - \left\{\frac{1}{6}u_{j-1,k-1} + u_{j-1,k} + \frac{1}{6}u_{j-1,k+1}\right\}\right] , \end{aligned}$$

$$\Delta u^*_{j,k-1} = \frac{1}{6}\left\{u_{j+1,k} + u_{j+1,k-1} - u_{j-1,k} - u_{j-1,k-1}\right\} .$$

The discretisation of $v\partial u/\partial y$ is obtained from (1) by replacing $(M_y, \Delta x)$ with $(M_x, \Delta y)$, (u^*, v) with (v^*, u) and reversing the role and orientation of (j,k). The discretisation of $u\partial u/\partial x$ and $v\partial v/\partial y$ then follow by direct substitution.

Equation (1) has been written deliberately in terms of u^* to bring out the structural similarity with finite difference discretisation. Group finite element discretisation could be obtained from (1) by replacing $u^*_{j+1,k+1}$ with $u_{j+1,k+1}$ and setting $\Delta u^*_{j,k+1} = 0$, etc. This gives an indication of the operation count comparison, and hence execution time comparison, for the group and conventional finite element methods.

Since the steady state solution is of interest, only the term, RHS^n, in (10.70), (10.71) has been modified. This requires changes to RHSBU; the relevant section of code replacing lines 17 to 30 is shown below. The present changes are invoked when $ME = 4$. To implement (1) in the existing RHSBU structure it is necessary to set $CX(2) = CX(1)$ and $CY(2) = CY(1)$ in TWBURG, to introduce UST, VST, to set $ES(1,2) = 0$ and to modify the evaluation of F and G, as indicated.

The u solution and accuracy for the conventional finite element method, AF-CON, are compared in the following table with those for the group finite element method (AF-MO, $\delta_x = \delta_y = 1/6$) for a 6x6 and an 11x11 grid. The conventional finite element is roughly twice as accurate and both methods are converging linearly with grid refinement. However the severity of the gradient in the x direction makes the accuracy of the solution at x=-0.20 responsible for most of the error. In addition the maximum gradient is effectively increasing as the grid is refined.

grid	scheme	x=-0.60	x=-0.20	x=0.20	x=0.60	rms error
6 x 6	AF-MO $\delta_x = \delta_y = 1/6$	1.0027	0.5231	-0.0327	-0.0252	0.0062
6 x 6	AF-CON	1.0012	0.5305	-0.0328	-0.0250	0.0030
11 x 11	AF-MO $\delta_x = \delta_y = 1/6$	0.9986	0.5271	-0.0338	-0.0250	0.0031
11 x 11	AF-CON	1.0000	0.5322	-0.0333	-0.0250	0.00155

Modifications to Subroutine RHSBU

```
      DO 6 K = 2,NYP
      IF(ME .NE. 4)GOTO 90
      KM = K-1
      KP = K+1
      UST(1,1) = (U(K,J)+0.5*(U(K,JM)+U(KM,JM))+U(KM,J))/3.
      UST(3,1) = (U(K,J)+0.5*(U(K,JM)+U(KP,JM))+U(KP,J))/3.
      UST(1,3) = (U(K,J)+0.5*(U(K,JP)+U(KM,JP))+U(KM,J))/3.
      UST(3,3) = (U(K,J)+0.5*(U(K,JP)+U(KP,JP))+U(KP,J))/3.
      UST(2,1) = 0.5*(U(K,J)+0.5*U(K,JM)+(U(KM,J)+U(KP,J))/6.
     1 + (U(KM,JM)+U(KP,JM))/12.)
      UST(2,3) = 0.5*(U(K,J)+0.5*U(K,JP)+(U(KM,J)+U(KP,J))/6.
     1 + (U(KM,JP)+U(KP,JP))/12.)
      UST(3,2) = (U(K,JP)+U(KP,JP)-U(K,JM)-U(KP,JM))/6.
      UST(1,2) = (U(K,JP)+U(KM,JP)-U(K,JM)-U(KM,JM))/6.
      UST(2,2) = 0.25*(((U(KM,JP)+U(KP,JP))/6.+U(K,JP))
     1                -((U(KM,JM)+U(KP,JM))/6.+U(K,JM)))
      VST(1,1) = (V(K,J)+0.5*(V(K,JM)+V(KM,JM))+V(KM,J))/3.
      VST(3,1) = (V(K,J)+0.5*(V(K,JM)+V(KP,JM))+V(KP,J))/3.
      VST(1,3) = (V(K,J)+0.5*(V(K,JP)+V(KM,JP))+V(KM,J))/3.
      VST(3,3) = (V(K,J)+0.5*(V(K,JP)+V(KP,JP))+V(KP,J))/3.
      VST(1,2) = 0.5*(V(K,J)+0.5*V(KM,J)+(V(K,JM)+V(K,JP))/6.
     1 + (V(KM,JM)+V(KM,JP))/12.)
      VST(3,2) = 0.5*(V(K,J)+0.5*V(KP,J)+(V(K,JM)+V(K,JP))/6.
     1 + (V(KP,JP)+V(KP,JM))/12.)
      VST(2,3) = (V(KP,J)+V(KP,JP)-V(KM,J)-V(KM,JP))/6.
      VST(2,1) = (V(KP,J)+V(KP,JM)-V(KM,J)-V(KM,JM))/6.
      VST(2,2) = 0.25*(((V(KP,JM)+V(KP,JP))/6.+V(KP,J))
     1               -((V(KM,JM)+V(KM,JP))/6.+V(KM,J)))
 90   RUD = 0.
```

```
      RVD = 0.
      DO 2 N = 1,3
      NK = K - 2 + N
      DO 1 M = 1,3
      MJ = J - 2 + M
      IF(ME .NE. 4)GOTO 100
      F(1) = UST(N,M)*U(NK,MJ)
      F(2) = UST(N,M)*V(NK,MJ)
      G(1) = VST(N,M)*U(NK,MJ)
      G(2) = VST(N,M)*V(NK,MJ)
      ES(1) = 0.
      ES(2) = 0.
      GOTO 120
 100  F(1) = U(NK,MJ)*U(NK,MJ)
      F(2) = U(NK,MJ)*V(NK,MJ)
      G(2) = V(NK,MJ)*V(NK,MJ)
      G(1) = F(2)
      ES(1) = 0.5*RE*U(NK,MJ)*(F(1) + G(2))
      ES(2) = 0.5*RE*V(NK,MJ)*(F(1) + G(2))
 120  DUM =   CX(M)*EMY(N)*F(1) + CY(N)*EMX(M)*G(1)
```

10.12 The listing of the program to generate the exact solutions of the two-dimensional Burgers' equation is given below. The program makes use of the subroutine EXBUR. Severe internal gradients are produced by the choice of the parameters given by (10.65). To produce moderate gradients in u adjacent to $x = 1$, a suitable parameter choice is given in Problem 10.14.

Listing of Code for Problem 10.12

```
C     PR10P12.
      PROGRAM CEXBURG
      DIMENSION A(5),UE(21,21),VE(21,21),PH(21,21)
      IMPLICIT  DOUBLE PRECISION (A-H,O-Z)
      OPEN(1,FILE='CEBURG.DAT')
      OPEN(6,FILE='CEBURG.OUT')
      READ(1,*)NX,NY,RE
      READ(1,*)(A(J),J=1,5),AL
      CALL EXBUR(UE,VE,A,AL,DX,DY,RE,NX,NY)
      DO 10 K=1,NY
   10 WRITE(6,2)K,(UE(J,K),J=1,NX)
      DO 11 K=1,NY
   11 WRITE(6,2)K,(VE(J,K),J=1,NX)
    2 FORMAT(I4,6F9.3)
      STOP
      END
```

10.13 Program TWBURG is run for 6×6, 11×11 and 21×21 grids and for various values of δ_x, δ_y, q_x and q_y. The computed rms solution and residual (rhs) errors are listed below. Unless otherwise stated the values of parameters for this problem are $a_1, a_2, = 1.3 \times 10^3, a_3 = a_4 = 0, a_5 = 1, \lambda = 25, x_0 = 1, \nu = 0.04$.

a) ME = 3, Optimal values of δ_x, δ_y

6 × 6 grid, (number of iterations to converge ≈ 11)

δ_x	δ_y	rms error	rhs error
0.26	0.26	0.0045	0.00120
0.16	0.16	0.0062	0.00170
0.1	0.1	0.0064	0.00170
0.06	0.06	0.0065	0.00180
0.0	0.16	0.0066	0.00180

11 × 11 grid, (number of iterations to converge ≈ 16) ($\Delta t = .0025$)

δ_x	δ_y	rms error	rhs error
0.26	0.26	0.00696	0.00189
0.16	0.16	0.00311	0.00083
0.1	0.1	0.00330	0.00087
0.0	0.16	0.00350	0.00092

21 × 21 grid, (number of iterations to converge ≈ 16)

δ_x	δ_y	rms error	rhs error
0.16	0.16	0.00110	0.00029
0.1	0.1	0.00125	0.00033
0.06	0.06	0.00133	0.00035
0.0	0.16	0.00143	0.00038

Thus the optimal values of δ_x and δ_y are seen to be $\delta_x = 0.26 = \delta_y$ for the $6 \times$ grid, $\delta_x = 0.16 = \delta_y$ for the 11×11 grid and the 21×21 grid.

b) ME=2, Optimal values of q_x and q_y

6 × 6 grid

q_x	q_y	rms error	rhs error
1.0	0.0	0.00542	0.00149
1.0	1.0	0.00627	0.00142
1.0	0.5	0.00576	0.00143
0.5	0.5	0.00630	0.00158
0.0	1.0	0.00717	0.00176

11 × 11 grid

q_x	q_y	rms error	rhs error
1.0	0.0	0.00196	0.00054
1.0	1.0	0.00213	0.00054
1.0	0.5	0.00202	0.00583
0.5	0.0	0.00267	0.00072

21 × 21 grid, ($\Delta t = 0.05$)

q_x	q_y	rms error	rhs error
1.0	0.0	0.000715	0.000198
1.0	1.0	0.000750	0.000194
1.0	0.5	0.000728	0.000195
0.5	0.0	0.000813	0.000221

The optimal values of q_x, q_y are found to be $q_x = 1$, $q_y = 0$ for all the grids used.

10.14 The program TWBURG is run for the given choice of parameters and an 11×11 grid.

a) ME=3, optimal values of δ_x, δ_y

δ_x	δ_y	rms error	rhs error
0.16	0.16	0.00147	0.000129
0.1	0.1	0.00198	0.000180
0.06	0.06	0.00224	0.000229
0.0	0.16	0.00255	0.000323
0.16	0.0	0.00148	0.000142

Thus $\delta_x = \delta_y = 0.16$ is again seen to be the optimal choice for this grid.

b) ME=2, optimal values of q_x, q_y

q_x	q_y	rms error	rhs error
1.0	0.0	0.00200	0.000356
1.0	1.0	0.00251	0.000288
1.0	0.5	0.00220	0.000296
0.5	0.0	0.00211	0.000135

$q_x = 0.5$, $q_y = 0$ is seen to be the optimal choice.

CTFD Solutions Manual: Chapter 11

Fluid Dynamics: The Governing Equations

11.1 The kinematic viscosity, ν, and the thermal diffusivity, α, depend on temperature, but not on pressure. As indicated in the figure, both ν and α have a similar dependence on temperature. This is to be expected since the Prandtl number, Pr $= \nu/\alpha$, and Table 11.1 indicates that the Prandtl number does not vary much with increasing temperature.

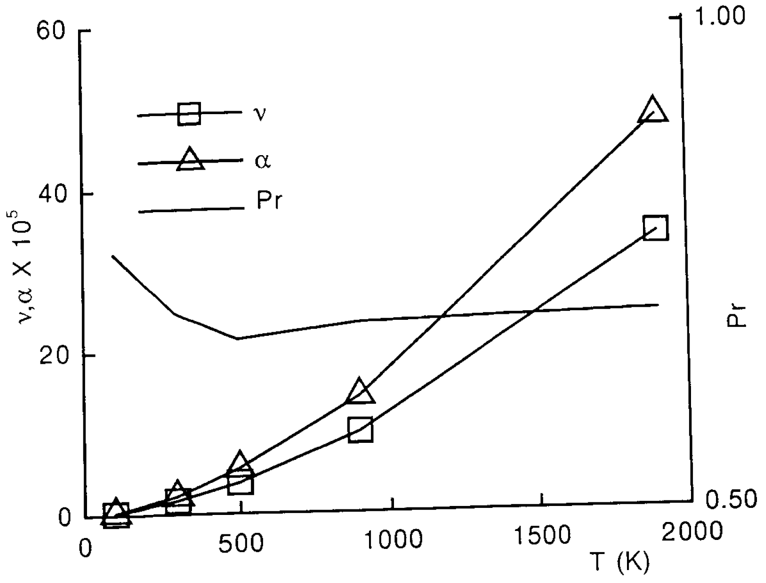

Fig. 1. Variation of ν,α and Pr for air with temperature.

11.2 The kinematic viscosity, ν, for water falls rapidly with increasing temperature up to $100°C$. For saturated conditions above this temperature the pressure is increasing rapidly and the kinematic viscosity falls at a much slower rate. The thermal diffusivity, α, rises slightly with increasing temperature up to $200°C$ and then diminishes. The Prandtl number falls with increasing temperature in a similar manner to ν.

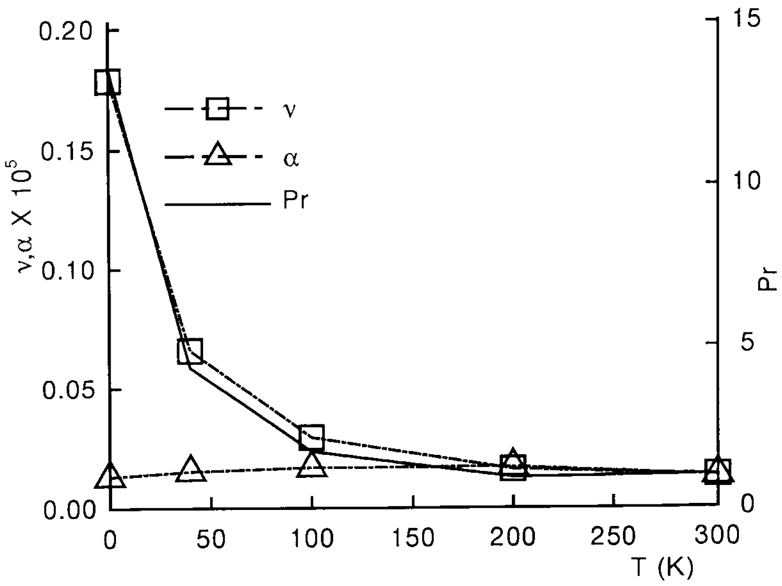

Fig. 1. (Problem 11.2) Variation of ν,α and Pr for water with temperature.

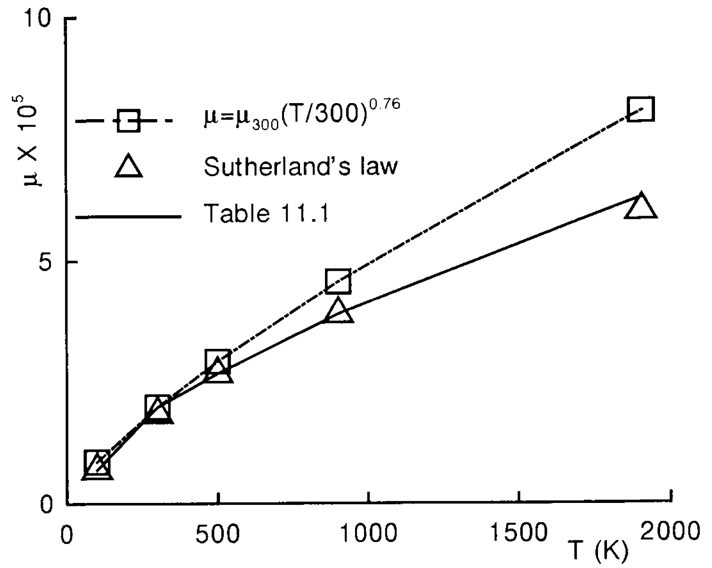

Fig. 1. (Problem 11.3) Variation of μ for air with temperature.

11.3 As is clear from the figure, Sutherland's law is a good approximation to the data for the viscosity, μ, given in Table 11.1. The power-law dependence on temperature is accurate for temperatures close to 300 K but diverges particularly for high temperatures.

11.4 We need to evaluate $\frac{\partial}{\partial t}\int_V \rho dV + \int_s \rho \mathbf{v}.\mathbf{n}\ ds = 0$ for a small element, $\{dr, rd\theta, dz\}$. The various parts are evaluated as

$$\frac{\partial}{\partial t}\int_V \rho dV = \frac{\partial \rho}{\partial t} rd\theta drdz$$

$$\int_s \rho \mathbf{v}.\mathbf{n}\ ds = \left\{\frac{\partial}{\partial r}(\rho v_r) + \frac{\rho v_r}{r} + \frac{1}{r}\frac{\partial}{\partial \theta}(\rho v_\theta) + \frac{\partial}{\partial z}(\rho v_z)\right\} rd\theta drdz.$$

$$\text{Thus}\quad \left\{\frac{\partial \rho}{\partial t} + \frac{1}{r}\frac{\partial}{\partial r}(\rho r v_r) + \frac{1}{r}\frac{\partial}{\partial \theta}(\rho v_\theta) + \frac{\partial}{\partial z}(\rho v_z)\right\} rd\theta drdz = 0.$$

As the element is shrunk to zero the required result is obtained.

11.5 For incompressible viscous flow, the global conservation of momentum equation can be written

$$\rho \int [\frac{D\mathbf{v}}{Dt} + \frac{1}{\rho}\nabla p]\ dV = \int \nabla.\underline{\tau}\ dV.$$

This is applied to a small volume element, $\{dr, rd\theta, dz\}$, with no circumferential variation ($\partial/\partial\theta = 0$) and for steady flow,

Axial momentum:

The left-hand side becomes

$$\rho\left\{v_z\frac{\partial v_z}{\partial z} + v_r\frac{\partial v_z}{\partial r} + \frac{1}{\rho}\frac{\partial p}{\partial z}\right\} rd\theta drdz.$$

The right-hand side becomes

$$\left\{\frac{1}{r}\frac{\partial}{\partial r}(r\tau_{rz}) + \frac{\partial}{\partial z}\tau_{zz}\right\} rd\theta drdz =$$

$$\mu\left\{\frac{\partial^2 v_z}{\partial r^2} + \frac{\partial^2 v_r}{\partial r \partial z} + \frac{1}{r}\left[\frac{\partial v_z}{\partial r} + \frac{\partial v_r}{\partial z}\right] + \frac{2\partial^2 v_z}{\partial z^2}\right\} rd\theta drdz.$$

Using continuity this simplifies to

$$RHS = \mu\{\nabla^2 v_z\} rd\theta drdz.$$

$$\text{Thus}\quad \left\{v_z\frac{\partial v_z}{\partial z} + v_r\frac{\partial v_z}{\partial r} + \frac{1}{\rho}\frac{\partial p}{\partial z} - \nu\ \nabla^2 v_z\right\} rd\theta drdz = 0.$$

As the volume element is shrunk to zero the required result is obtained.

Radial momentum:

The left-hand side becomes

$$\rho\left\{v_z\frac{\partial v_r}{\partial z}+v_r\frac{\partial v_r}{\partial r}+\frac{1}{\rho}\frac{\partial p}{\partial r}\right\}rd\theta drdz.$$

The right-hand side becomes

$$\left\{\frac{1}{r}\frac{\partial}{\partial r}(r\tau_{rr})+\frac{\partial}{\partial z}\tau_{zr}-\frac{2\mu v_r}{r^2}\right\}rd\theta drdz$$

$$=\mu\left\{\frac{2}{r}\frac{\partial}{\partial r}\left(\frac{r\partial v_r}{\partial r}\right)+\frac{\partial^2 v_z}{\partial r\partial z}+\frac{\partial^2 v_r}{\partial z^2}-\frac{2v_r}{r^2}\right\}rd\theta drdz.$$

This can be simplified by using the continuity equation,

$$RHS=\mu\left\{\frac{\partial^2 v_r}{\partial r^2}+\frac{1}{r}\frac{\partial v_r}{\partial r}+\frac{\partial^2 v_r}{\partial z^2}-\frac{v_r}{r^2}\right\}rd\theta drdz.$$

Combining the above results gives

$$\left\{v_r\frac{\partial v_r}{\partial r}+v_z\frac{\partial v_r}{\partial z}+\frac{1}{\rho}\frac{\partial p}{\partial r}-\nu\nabla^2 v_r+\nu\frac{v_r}{r^2}\right\}rd\theta drdz=0.$$

For a zero size of the volume element the required result is obtained.

11.6 The energy equation, (11.32), can be written for steady incompressible flow as

$$\int_V\left(\rho\frac{De}{Dt}-\Phi-k\,\nabla^2T\right)dV=0.$$

This follows from considering the integral form of (11.35) with (11.37) substituted. The desired result follows, for any coordinate system, by shrinking the volume to zero.

The form of $\nabla^2 T$ in spherical coordinates is given by Panton (1984), *p.*744.

For steady flow, $De/Dt=\nabla.(\mathbf{v}e)-e\;\nabla.\mathbf{v}$,

where $\nabla.\mathbf{v}=0$ for incompressible flow. Thus

$$\begin{aligned}\frac{De}{Dt}&=\frac{1}{r^2}\frac{\partial}{\partial r}(r^2v_re)+\frac{1}{r\sin\theta}\frac{\partial}{\partial\theta}(v_\theta\sin\theta\;e)+\frac{1}{r\sin\theta}\frac{\partial(v_\phi e)}{\partial\phi}\\&-e\left\{\frac{1}{r^2}\frac{\partial}{\partial r}(r^2v_r)+\frac{1}{r\sin\theta}\frac{\partial}{\partial\theta}(v_\theta\sin\theta)+\frac{1}{r\sin\theta}\frac{\partial}{\partial\phi}v_\phi\right\}\\&=v_r\frac{\partial e}{\partial r}+\frac{v_\theta}{r}\frac{\partial e}{\partial\theta}+\frac{v_\phi}{r\sin\theta}\frac{\partial e}{\partial\phi}.\end{aligned}$$

11.7 To obtain the same form of the momentum equations in nondimensional variables it is necessary to define nondimensional viscous stresses as $\tau^*=\tau/\rho_\infty U_\infty^2$. Thus τ_{xx} becomes

$$\rho_\infty U_\infty^2\tau_{xx}^*=\mu_\infty\frac{U_\infty}{L}\left\{\frac{-2}{3}\mu^*\left(\frac{\partial u^*}{\partial x^*}+\frac{\partial v^*}{\partial y^*}\right)+2\mu^*\frac{\partial u^*}{\partial x^*}\right\}$$

$$\text{or}\quad\tau_{xx}^*=\frac{\mu^*}{Re}\left\{2\frac{\partial u^*}{\partial x^*}-\frac{2}{3}\mathcal{D}^*\right\}.$$

To obtain the same form of the energy equation in nondimensional variables it is necessary to define nondimensional heat transfer rates as $\dot{Q}^*=\dot{Q}/\rho_\infty U_\infty^3$. Thus $\dot{Q}_x$ becomes

$$\rho_\infty U_\infty^3\dot{Q}_x^*=-k_\infty k^*\frac{T_\infty}{L}\frac{\partial T^*}{\partial x^*}$$

$$\text{or}\quad\dot{Q}_x^*=-k^*\frac{\partial T^*}{\partial x^*}/\{(\gamma-1)M_\infty^2Pr\;\text{Re}\}\;.$$

a) The nondimensionalisation is suitable for high-speed flow where temperature changes are caused by compressibility.

b) For low speed flow it is better to nondimensionalise the temperature as $T^*=(T-T_w)/(T_\infty-T_w)$ and heat transfer rates as $\dot{Q}^*=\dot{Q}/\{\rho_\infty U_\infty c_p(T_\infty-T_w)\}$.

11.8 **a)** For a stream function approach typical boundary conditions are:

(i) solid surface: $\psi=0$
(ii) far field: $\partial\psi/\partial y=U_\infty$.

Often the farfield (upstream) boundary condition can be converted into an equivalent Dirichlet boundary condition. As a result the stream function formulation may produce an algorithm that converges more rapidly than a velocity potential approach.

b) The velocity potential formulation extends directly to three dimensions, (11.51). However the stream function formulation is based on satisfying the continuity equation automatically in two dimensions. But this is not possible in three dimensions and the scalar stream function must be replaced with a three-component vector potential (Sect. 11.5.1).

11.9 For potential flow around a circular cylinder, (11.58) and (11.59) can be combined at the surface of the cylinder to give a tangential velocity component,

$$q_T=2U_\infty\sin\theta.$$

In addition, the Bernoulli equation, (11.48), gives

$$p_\infty/\rho+0.5\;U_\infty^2=p/\rho+0.5\;q_T^2\;.$$

Thus $c_p=(p-p_\infty)/0.5\rho U_\infty^2=1-(q_T/U_\infty)^2=1-4\sin^2\theta$.

11.10 We define $\psi=Pfu_e^{1/2}$, $P^2=(2-\beta)x\nu$ and $\eta=u_e^{1/2}y/P$.

The introduction of the stream function automatically satisfies the continuity equation. The various terms in the x-momentum equation become

$u=u_ef'$ where $f'\equiv\partial f/\partial\eta$, $du_e/dx=\{\beta/(2-\beta)\}u_e/x$

$$\frac{\partial u}{\partial x}=\frac{\beta}{(2-\beta)}\frac{u_e}{x}f'+(\frac{\beta-1)}{(2-\beta)}\frac{u_e^{3/2}}{x}\frac{y}{P},\quad \frac{\partial u}{\partial y}=\frac{u_e^{3/2}}{P}f'',\quad \frac{\partial^2 u}{\partial y^2}=\frac{u_e^2}{P^2}f'''$$

$$v=\frac{u_e^{1/2}}{(2-\beta)x}[(1-\beta)u_e^{1/2}yf'-Pf].$$

Substitution into the x-momentum equation gives

$$\frac{u_e^2}{x(2-\beta)}[\beta(f')^2-ff'']=\frac{u_e^2}{x(2-\beta)}[\beta+f''']$$

or $\quad f'''+ff''+\beta\{1-(f')^2\}=0.$

11.11 It is convenient to write (11.61) as

$$\frac{\partial}{\partial x}(u^2)+\frac{\partial}{\partial y}(uv)-u_e\frac{du_e}{dx}=\nu\frac{\partial^2 u}{\partial y^2}.$$

After integrating across the boundary layer ($h>\delta$)

$$\int_o^h[\frac{\partial}{\partial x}(u^2)-u_e\frac{du_e}{dx}]dy+u_ev_h=-\tau_w/\rho.$$

From the continuity equation $\quad u_ev_h=-\int_o^h u_e\frac{\partial u}{\partial x}dy.$

Thus,

$$\int_o^h\left\{\frac{\partial}{\partial x}[u(u-u_e)]+(u-u_e)\frac{du_e}{dx}\right\}dy=-\tau_w/\rho.$$

But $\quad u=u_e \quad$ if $y>\delta$

$$\text{Thus}\quad \frac{d}{dx}\left[u_e^2\int_o^\delta\frac{u}{u_e}\left(1-\frac{u}{u_e}\right)dy\right]+u_e\frac{du_e}{dx}\int_o^\delta\left(1-\frac{u}{u_e}\right)dy=\tau_w/\rho,$$

$$\text{or}\quad \frac{1}{u_e^2}\frac{d}{dx}(u_e^2\theta)+\frac{\delta^*}{u_e}\frac{du_e}{dx}=0.5\,c_f,$$

$$\text{or}\quad \frac{d\theta}{dx}=0.5\,c_f-(H+2)\frac{\theta}{u_e}\frac{du_e}{dx}.$$

Exactly the same equation would be obtained starting from the turbulent boundary layer equations, (11.73) and (11.74).

11.12 From the definition of the Bernoulli variable, H,

$$\partial H/\partial x=\partial p/\partial x+u\partial u/\partial x+v\partial v/\partial x \tag{1}$$

$$\partial H/\partial y=\partial p/\partial y+u\partial u/\partial y+v\partial v/\partial y\ . \tag{2}$$

From the definition of vorticity,

$$\partial\zeta/\partial x=\partial^2u/\partial x\partial y-\partial^2v/\partial x^2.$$

Also using continuity, $\quad \partial\zeta/\partial x=-(\partial^2v/\partial x^2+\partial^2v/\partial y^2)\ .$ (3)

Similarly, $\quad \partial\zeta/\partial y=(\partial^2u/\partial x^2+\partial^2u/\partial y^2)\ .$ (4)

Using (1) and (4), the steady counterpart of (11.83) becomes

$$v\partial u/\partial y+\partial H/\partial x-v\partial v/\partial x=(1/Re)\partial\zeta/\partial y$$

$$\text{or}\quad v\zeta+\partial H/\partial x=(1/Re)\partial\zeta/\partial y. \tag{5}$$

Using (2) and (3), the steady counterpart of (11.84) becomes

$$-u\zeta+\partial H/\partial y=-(1/Re)\partial\zeta/\partial x. \tag{6}$$

In the farfield equations (5) and (6) reduce to H = constant. Therefore the farfield could be solved as a solution of (11.51) with embedded constants determined by matching the solution of (5) and (6) and the continuity equation in the viscous region close to the surface.

11.13 $\partial(11.83)/\partial x+\partial(11.84)/\partial y$ becomes

$$\frac{\partial}{\partial t}(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y})+(\frac{\partial u}{\partial x})^2+2\frac{\partial u}{\partial y}\frac{\partial v}{\partial x}+(\frac{\partial v}{\partial y})^2+\frac{\partial^2 p}{\partial x^2}+\frac{\partial^2 p}{\partial y^2}$$
$$=\frac{1}{Re}\{\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}\}\{\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\}=0$$

$$\frac{\partial^2 p}{\partial x^2}+\frac{\partial^2 p}{\partial y^2}=-2(\frac{\partial^2\psi}{\partial x\partial y})^2+2\frac{\partial^2\psi}{\partial x^2}\frac{\partial^2\psi}{\partial y^2}.$$

In three dimensions an equivalent treatment leads to

$$\frac{\partial^2 p}{\partial x^2}+\frac{\partial^2 p}{\partial y^2}+\frac{\partial^2 p}{\partial z^2}$$
$$=-\{\frac{\partial^2(u^2)}{\partial x^2}+2\frac{\partial^2(uv)}{\partial x\partial y}+\frac{\partial^2(v^2)}{\partial y^2}+2\frac{\partial(vw)}{\partial y\partial z}+\frac{\partial^2(w^2)}{\partial z^2}+2\frac{\partial^2(uw)}{\partial x\partial z}\}.$$

11.14 Equation (11.101) can be rewritten, for an isentropic process, as

$$udu+vdv+a^2d\rho/\rho=0. \tag{1}$$

Equation (11.10) can be expanded to give

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{u}{\rho}\frac{\partial\rho}{\partial x}+\frac{v}{\rho}\frac{\partial\rho}{\partial y}=0.$$

Using (1), this becomes

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}-\frac{u^2}{a^2}\frac{\partial u}{\partial x}-\frac{uv}{a^2}\frac{\partial v}{\partial x}-\frac{v^2}{a^2}\frac{\partial v}{\partial y}-\frac{uv}{a^2}\frac{\partial u}{\partial y}=0.$$

Using (11.102) this can be re-arranged to give

$$(\frac{u^2}{a^2}-1)\frac{\partial^2\phi}{\partial x^2}+(\frac{v^2}{a^2}-1)\frac{\partial^2\phi}{\partial y^2}+2\frac{uv}{a^2}\frac{\partial^2\phi}{\partial x\partial y}=0\ ,$$

which is the two-dimensional equivalent of (11.103) .

11.15 The following orders-of-magnitude are ascribed to various terms,

$u \approx u_e,\quad v\approx \delta u_e/L,\quad T\approx T_w,\quad \partial/\partial x\approx 1/L,\quad \partial/\partial y \approx 1/\delta$

From the y-momentum equation, $\partial p/\partial y \approx u_e^2\delta/L,$

so that $v\partial p/\partial y << u\partial p/\partial x$ *and* $\partial p/\partial x \approx p_e/L.$

The biggest term in Φ is $\mu(\partial u/\partial y)^2\approx \mu u_e^2/\delta^2.$

The heat conduction terms are of magnitude

$$\frac{\partial}{\partial x}\{k\frac{\partial T}{\partial x}\}\approx\frac{kT_w}{L^2};\quad \frac{\partial}{\partial y}\{\frac{k\partial T}{\partial y}\}\approx\frac{kT_w}{\delta^2}$$

Consequently $\partial\{k\partial T/\partial x\}/\partial x$ can be discarded.

All other terms must be retained. Thus for air, (11.38) becomes

$$\rho c_p\{u\frac{\partial T}{\partial x}+v\frac{\partial T}{\partial y}\}=u\frac{dp_e}{dx}+\frac{\partial}{\partial y}\{k\frac{\partial T}{\partial y}\}+\mu\{\frac{\partial u}{\partial y}\}^2$$

which is (11.113).

11.16 The x-momentum equation (part of 11.116) becomes

$$\frac{\partial}{\partial t}(\rho u)+\frac{\partial}{\partial x}(\rho u^2+p)+\frac{\partial}{\partial y}(\rho uv)=\frac{\partial}{\partial x}\tau_{xx}+\frac{\partial}{\partial y}\tau_{xy}$$
$$=\frac{\partial}{\partial x}\{\mu(\frac{4}{3}\frac{\partial u}{\partial x}-\frac{\partial v}{\partial y})\}+\frac{\partial}{\partial y}\{\mu(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x})\},$$

and the y-momentum equation becomes

$$\frac{\partial}{\partial t}(\rho v)+\frac{\partial}{\partial y}(\rho v^2+p)+\frac{\partial}{\partial x}(\rho uv)=\frac{\partial}{\partial x}\tau_{xy}+\frac{\partial}{\partial y}\tau_{yy}$$
$$=\frac{\partial}{\partial x}\{\mu(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x})\}+\frac{\partial}{\partial y}\{\mu(\frac{4}{3}\frac{\partial v}{\partial y}-\frac{\partial u}{\partial x})\}.$$

11.17 The x and y momentum equations (in 11.116) can be combined as $u(x\text{-mmtm})+v(y\text{-mmtm})$, with the help of the continuity equation, to give

$$\frac{\partial}{\partial t}(0.5\rho q^2)+\frac{\partial}{\partial x}(0.5\rho uq^2)+\frac{\partial}{\partial y}(0.5\rho vq^2)+u\partial p/\partial x+v\partial p/\partial y$$
$$-\{u\frac{\partial}{\partial x}\tau_{xx}+u\frac{\partial}{\partial y}\tau_{xy}+v\frac{\partial}{\partial x}\tau_{xy}+v\frac{\partial}{\partial y}\tau_{yy}\}=0. \qquad (1)$$

This is equivalent to (11.34) with $\mathbf{f}=0$.

Subtracting (1) from the two-dimensional equivalent of (11.33) gives

$$c_v\{\frac{\partial}{\partial t}(\rho T)+\frac{\partial}{\partial x}(\rho uT)+\frac{\partial}{\partial y}(\rho vT)\}+p\{\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\}$$
$$=\Phi+\frac{\partial}{\partial x}\{\frac{k\partial T}{\partial x}\}+\frac{\partial}{\partial y}\{\frac{k\partial T}{\partial y}\}, \qquad (2)$$

which is the equivalent of (11.35) after substituting (11.37).

It may be noted that $\rho c_v T=\rho c_p T-p.$ Thus (2) can be rewritten as

$$c_p\{\frac{\partial}{\partial t}(\rho T)+\frac{\partial}{\partial x}(\rho uT)+\frac{\partial}{\partial y}(\rho vT)\}-\frac{Dp}{Dt}$$
$$=\Phi+\frac{\partial}{\partial x}\{\frac{k\partial T}{\partial x}\}+\frac{\partial}{\partial y}\{\frac{k\partial T}{\partial y}\},$$

which is the equivalent of (11.36).

CTFD Solutions Manual: Chapter 12

Generalised Curvilinear Coordinates

12.1 For a two-dimensional transformation we have

$$\underline{J} = \begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} \quad \text{and} \quad \underline{J}^{-1} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix}, \tag{1}$$

also $\underline{JJ}^{-1} = I$.

Thus multiplying $\underline{J}$ and $\underline{J}^{-1}$ we get

$$\begin{bmatrix} \xi_x x_\xi + \xi_y y_\xi & \xi_x x_\eta + \xi_y y_\eta \\ \eta_x x_\xi + \eta_y y_\xi & \eta_x x_\eta + \eta_y y_\eta \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{2}$$

Now solving (2) for ξ_x, ξ_y, η_x and η_y, we obtain the two-dimensional version of (12.7).

12.2 In (12.13),

$$\begin{aligned} LHS = |\mathbf{g}|^{\frac{1}{2}} &= \{(x_\xi^2 + y_\xi^2)(x_\eta^2 + y_\eta^2) - (x_\xi x_\eta + y_\xi y_\eta)^2\}^{\frac{1}{2}} \\ &= \pm(x_\eta y_\xi - x_\xi y_\eta) , \end{aligned} \tag{1}$$

$$RHS = |\underline{J}^{-1}| = x_\xi y_\eta - x_\eta y_\xi \qquad \text{(from 12.4)} , \tag{2}$$

$LHS = RHS$ as required.

12.3 Considering first the term x_ξ, we have from (12.15)

$$x_\xi = (g_{11})^{\frac{1}{2}} \cos\alpha \, . \tag{1}$$

From (12.17), $\cos\theta = g_{12}/(g_{11}\, g_{22})^{\frac{1}{2}}$, (2)

$$\text{i.e.} \quad \sin\theta = \left(1 - \frac{g_{12}^2}{g_{11}g_{22}}\right)^{\frac{1}{2}} = \left(\frac{g_{11}g_{22} - g_{12}^2}{g_{11}\, g_{22}}\right)^{\frac{1}{2}} . \tag{3}$$

Since $|\mathbf{g}|^{\frac{1}{2}} = |\underline{J}^{-1}|$ (see 12.13), and $|g| = g_{11}\, g_{22} - g_{12}^2$, equation (3) can be written as, (from 12.16)

$$\sin\theta = \frac{|\underline{J}^{-1}|}{(g_{11}\, g_{22})^{\frac{1}{2}}} = \frac{|\underline{J}^{-1}|}{g_{11}} \left(\frac{g_{11}}{g_{22}}\right)^{\frac{1}{2}} = \frac{|\underline{J}^{-1}|}{g_{11}} \frac{1}{(AR)} \, . \tag{4}$$

$$\text{Thus,} \quad g_{11} = \frac{|\underline{J}^{-1}|}{AR.\sin\theta} = \frac{1}{AR.J.\sin\theta} , \tag{5}$$

so that (1) can be written as,

$x_\xi = \cos\alpha/(AR.J.\sin\theta)^{\frac{1}{2}}$ as required.

The expression, $y_\xi = -\sin\alpha/(AR.J.\sin\theta)^{\frac{1}{2}}$

can be derived as follows.

$$g_{11} = x_\xi^2 + y_\xi^2 \quad \text{and} \quad y_\xi = (g_{11} - x_\xi^2)^{\frac{1}{2}} \, . \tag{6}$$

Upon substituting for g_{11} and x_ξ, the required expression for y_ξ is obtained.

Further, from (4) and (5) we get $AR/(J\sin\theta) = g_{22}$. (7)

From (5) and (6), $\sin\alpha = y_\xi/\sqrt{g_{11}}$, $\cos\alpha = x_\xi/\sqrt{g_{11}}$. (8)

Now expanding cos $(\theta - \alpha)$ and sin $(\theta - \alpha)$ and substituting for sinα, cosα, sinθ and cosθ from the previous expressions, one obtains

$$x_\eta = \cos(\theta - \alpha)\left(\frac{AR}{J\sin\theta}\right)^{\frac{1}{2}} \quad \text{and} \quad y_\eta = \sin(\theta - \alpha)\left(\frac{AR}{J\sin\theta}\right)^{\frac{1}{2}}$$

as required.

12.4 From (12.7), $\xi_x = y_\eta/|\underline{J}^{-1}|$. (1)

Using the chain rule, $\xi_{xx} = \xi_x(\partial\xi_x/\partial\xi) + \eta_x(\partial\xi_x/\partial\eta)$

$$\text{i.e.} \quad \xi_{xx} = \xi_x \left\{ \frac{|J^{-1}|y_{\xi\eta} - y_\eta|J^{-1}|_\xi}{(J^{-1})^2} \right\} + \eta_x \left\{ \frac{|J^{-1}|y_{\eta\eta} - y_\eta|J^{-1}|_\eta}{(J^{-1})^2} \right\} .$$

Now substituting $\xi_x|J^{-1}|$ for y_η in the numerator and simplifying the form of expression given for ξ_{xx} is obtained. The procedure is similar for ξ_{xy}.

12.5 The RHS of (12.45) is given by

$$\frac{T_\xi}{x_\xi}\left[1 + \frac{\Delta\xi^2}{6}\left(\frac{T_{\xi\xi\xi}}{T_\xi} - \frac{x_{\xi\xi\xi}}{x_\xi}\right) + ...\right] . \tag{1}$$

From (12.35) we have $T_\xi = x_\xi T_x$. (2)

Hence (1) becomes,

$$T_x + T_x\frac{\Delta\xi^2}{6}\left(\frac{T_{\xi\xi\xi}}{T_\xi} - \frac{x_{\xi\xi\xi}}{x_\xi}\right) + ... \tag{3}$$

Using the chain rule, from (2), we have

$$T_{\xi\xi} = x_{\xi\xi}T_x + x_\xi^2 T_{xx} \quad \text{and} \quad T_{\xi\xi\xi} = T_x x_{\xi\xi\xi} + 3x_\xi x_{\xi\xi} T_{xx} + x_\xi^3 T_{xxx} \ . \tag{4}$$

Substituting (4) in (3) and simplifying (12.46) is obtained.

12.6 Expanding the terms on the LHS of the given expression as a Taylor series about 'j' we have

$$\frac{T_{j+1}-T_{j-1}}{x_{j+1}-x_{j-1}} =$$

$$\frac{(1+r_\xi)\Delta\xi T_\xi + (r_\xi^2-1)\frac{\Delta\xi^2}{2}T_{\xi\xi} + (r_\xi^3+1)\frac{\Delta\xi^3}{6}T_{\xi\xi\xi} + O(H)}{(1+r_\xi)\Delta\xi x_\xi + (r_\xi^2-1)\frac{\Delta\xi^2}{2}x_{\xi\xi} + (r_\xi^3+1)\frac{\Delta\xi^3}{6}x_{\xi\xi\xi} + O(H)}$$

$$= \frac{T_\xi}{x_\xi}\left[1 + (r_\xi - 1)\frac{\Delta\xi}{2}\frac{T_{\xi\xi}}{T_\xi} + \frac{(r_\xi^3+1)}{(r_\xi+1)}\frac{\Delta\xi^2}{6}\frac{T_{\xi\xi\xi}}{T_\xi}\ \ldots\right]$$

$$\Big/ \left[1 + (r_\xi - 1)\frac{\Delta\xi}{2}\frac{x_{\xi\xi}}{x_\xi} + \frac{(r_\xi^3+1)}{(r_\xi+1)}\frac{\Delta\xi^2}{6}\frac{x_{\xi\xi\xi}}{x_\xi}\ \ldots\right]$$

$$= \frac{T_\xi}{x_\xi}\left[1 + (r_\xi - 1)\frac{\Delta\xi}{2}\left\{\frac{T_{\xi\xi}}{T_\xi} - \frac{x_{\xi\xi}}{x_\xi}\right\} + \left(\frac{r_\xi^3+1}{r_\xi+1}\right)\frac{\Delta\xi^2}{6}\left\{\frac{T_{\xi\xi\xi}}{T_\xi} - \frac{x_{\xi\xi\xi}}{x_\xi}\right\}\ldots\right] . \tag{1}$$

But,

$$\frac{T_\xi}{x_\xi}\left\{\frac{T_{\xi\xi}}{T_\xi} - \frac{x_{\xi\xi}}{x_\xi}\right\} = x_\xi T_{xx} \ ,$$

$$\frac{T_\xi}{x_\xi} = T_x \quad , \quad \frac{T_\xi}{x_\xi}\left(\frac{T_{\xi\xi\xi}}{T_\xi} - \frac{x_{\xi\xi\xi}}{x_\xi}\right) = x_\xi^2 T_{xxx} + 3x_{\xi\xi}T_{xx} .$$

Hence

$$\frac{T_{j+1}-T_{j-1}}{x_{j+1}-x_{j-1}} = T_x + (r_\xi - 1)\frac{\Delta\xi}{2}T_{xx}\frac{(1+r_x)}{(1+r_\xi)}\frac{\Delta x}{\Delta\xi} +$$

$$\left(\frac{r_\xi^2+1}{r_\xi+1}\right)\frac{\Delta\xi^2}{6}\{x_\xi^2 T_{xxx} + 3x_{\xi\xi}T_{xx}\},$$

which leads to the required result.

From the resulting expression we find that second order accuracy is achieved for all

$$r_x = \left(\frac{3 - r_\xi^2 + 2r_\xi^3}{3r_\xi^3 - r_\xi + 2}\right) .$$

12.7 Following the transformation carried out in Sect. 12.3.1, we find that the given equations,

i.e. $\quad u_x + v_y = 0 \quad$ and $\quad u_y - v_x = 0$ (1)

may be written in generalised coordinates as

$$(1/J)(\xi_x u_\xi + \eta_x u_\eta + \xi_y v_\xi + \eta_y v_\eta) = 0$$

and $\quad (1/J)(\xi_y u_\xi + \eta_y u_\eta - \xi_x v_\xi - \eta_x v_\eta) = 0$. (2)

Now writing $\xi_x u_\xi / J = (\xi_x u/J)_\xi - u\,(\xi_x/J)_\xi = (\xi_x u/J)_\xi - u(y_n)_\xi$ (see 12.50) etc.

Equation (2) reduces to

$U_\xi^* + V_\eta^* = 0 \quad$ and $\quad U_\eta^* - V_\xi^* = 0$, (3)

where $U^* = (1/J)(\xi_x u + \xi_y v)$, and $V^* = (1/J)(\eta_x u + \eta_y v)$. (4)

Now $\xi_x = y_\eta / |\underline{J}^{-1}|$ from (12.7).

Since the grid is conformal, $y_\eta = x_\xi$, so that

$\xi_x = x_\xi / |\underline{J}^{-1}| = \eta_y |\underline{J}^{-1}| / |\underline{J}^{-1}| = \eta_y \quad$ from (12.7).

As a result in (4), we have

$U^* = (1/J)(\eta_y u + \xi_y v)$.

Similarly $\eta_x = -\xi_y$ for a conformal grid and $V^* = (1/J)(\eta_y v - \xi_y u)$.

12.8 Following the procedure for the continuity equation in Sect. 12.8.3 and the Laplace equation in Sect. 12.1.3, one can write the vorticity transport equation in generalised coordinates as

$$\left(\frac{\zeta U^c}{J}\right)_\xi + \left(\frac{\zeta V^c}{J}\right)_\eta - \frac{1}{Re}\left(\frac{g_{22}}{g^{\frac{1}{2}}}\zeta_\xi - \frac{g_{12}}{g^{\frac{1}{2}}}\zeta_\eta\right)_\xi - \frac{1}{Re}\left(-\frac{g_{12}}{g^{\frac{1}{2}}}\zeta_\xi + \frac{g_{11}}{g^{\frac{1}{2}}}\zeta_\eta\right)_\eta = 0 , \tag{1}$$

where $g_{22} = x_\eta^2 + y_\eta^2$, $\quad g_{11} = x_\xi^2 + y_\xi^2$, $\quad g_{12} = x_\xi x_\eta + y_\xi y_\eta$.

For an orthogonal grid, $g_{12} = 0$, so that (1) becomes

$$\left(\frac{\zeta U^c}{J}\right)_\xi + \left(\frac{\zeta V^c}{J}\right)_\eta - \frac{1}{Re}\left(\frac{g_{22}}{g^{\frac{1}{2}}}\zeta_\xi\right)_\xi - \frac{1}{Re}\left(\frac{g_{11}}{g^{\frac{1}{2}}}\zeta_\eta\right)_\eta = 0 , \tag{2}$$

i.e.

$$\left(\frac{\zeta U^c}{J}\right)_\xi + \left(\frac{\zeta V^c}{J}\right)_\eta - \frac{1}{Re}\left(\frac{x_\eta^2 + y_\eta^2}{|\underline{J}^{-1}|}\zeta_\xi\right)_\xi - \frac{1}{Re}\left(\frac{x_\xi^2 + y_\xi^2}{|\underline{J}^{-1}|}\zeta_\eta\right)_\eta = 0 . \tag{3}$$

For a conformal grid,

$$\left(\frac{\zeta U^c}{J}\right)_\xi + \left(\frac{\zeta V^c}{J}\right)_\eta - \frac{1}{J.Re}[\zeta_{\xi\xi} + \zeta_{\eta\eta}] = 0 . \tag{4}$$

12.9 Following the procedure in the previous problems, the given equations are transformed into generalised coordinates as follows,

$$\left(\frac{U^c}{J}\right)_\xi + \left(\frac{V^c}{J}\right)_\eta = 0 \,, \tag{1}$$

and $u\{\xi_x u_\xi + \eta_x u_\eta\} + v\{\xi_y u_\xi + \eta_y u_\eta\} + p_{ex}$

$$= \frac{\xi_y}{Re}\{(1+\nu_T)\frac{\partial u}{\partial y}\}_\xi + \frac{\eta_y}{Re}\{(1+\nu_T)\frac{\partial u}{\partial y}\}_\eta \,. \tag{2}$$

Equation (2) can be expanded as

$$U^c u_\xi + V^c u_\eta + p_{ex} = \frac{1}{Re}\Big\{ \big(\xi_y(1+\nu_T)_\xi + \eta_y(1+\nu_T)_\eta\big)\big(\xi_y u_\xi + \eta_y u_\eta\big) + \\ (1+\nu_T)\big(\xi_y^2 u_{\xi\xi} + 2\xi_y\eta_y u_{\xi\eta} + \eta_y^2 u_{\eta\eta}\big)\Big\}. \tag{3}$$

In the momentum equation i.e. (3), under the boundary layer approximation some of the ξ-derivatives can be dropped. Accordingly, (3) reduces to

$$U^c u_\xi + V^c u_\eta + p_{ex} \quad = \quad \frac{\eta_y^2}{Re}\{(1+\nu_T)u_\eta\}_\eta . \tag{4}$$

12.10 The number of iterations to converge (N_i) and the solution error ($\|\theta - \theta_{ex}\|_{rms}$) using LAGEN and using FIVOL for the given computational domain are as follows. The values of the various parameters are: $r_w = r_z = 0.1$, $r_x = 1.0$, $r_y = 3$, $\theta_{wx} = 0$, $\theta_{zy} = 90^0$, $\lambda = 1.5$.

	Grid	N_i	error
LAGEN	6×6	16	0.1066
	11×11	20	0.04742
	21×21	38	0.03388
FIVOL	6×6	16	0.3950
	11×11	24	0.1708
	21×21	>51	0.05807

It is clearly seen that as the mesh gets more distorted the accuracy of LAGEN (i.e. the generalised coordinates method) worsens. The finite-volume code FIVOL has a similar behaviour. With a fine grid the accuracies of LAGEN and FIVOL approach each other (compare the solution errors for the 21×21 grid).

12.11 The grid for this problem is generated by using the expressions suggested. Thus

$$r = r_{wz} + a_1 s + a_2 s^2 \text{ where } 0 \le s \le 1 \text{ as } \quad r_{wz} \le r \le r_{xy} \,. \tag{1}$$

a_1 and a_2 are evaluated using the boundary conditions that

$r = r_{xy}$ when $s = 1$ and

$r = r_{wz} + \Delta r_{wz}$ when $s = 1/JMAX$. We find that

$$a_1 = \Delta r_{wz} - ss^2(r_{xy} - r_{wz})/ss(1-ss) \,,$$

$$a_2 = r_{xy} - r_{wz} - a_1 \,, \tag{2}$$

where $ss = (JMAX-1)/JMAX$.

Similarly in the θ direction,

$\theta = \theta_b + b_1 t + b_2 t^2$ and

$\theta = \theta_n$ when $t = 1$ and $\theta = \theta_n - \Delta\theta_n$ when $t = (KMAX-1)/KMAX$.
we find that

$$b_1 = \big((\theta_n - \theta_b - \Delta\theta_n) - (\theta_n - \theta_b)tt^2\big)/\big(tt(1-tt)\big) \,, \tag{3}$$

$$b_2 = \theta_n - \theta_b - b_1$$

and $tt = (KMAX-1)/KMAX$. (4)

The values of Δr_{wz} and $\Delta\theta_n$ are given. A modified version of subroutine GRID is developed to implement (1) to (4) and its listing is provided below. Note that Δr_{wz} and $\Delta\theta_n$ are read in. On running the modified program for the conditions of Table 12.2, the following computed errors are obtained. Here the values of Δr_{wz} and $\Delta\theta_n$ were 0.02 and 4.0 respectively.

	grid	error
$r_x = 1$	6×6	1.621
	11×11	0.1443
	21×21	0.00688
$r_x = 2$	21×21	0.01232

The accuracy is seen to improve considerably on the fine grid when compared with that for a uniform mesh (Table 12.2). It should be pointed out that the accuracy is quite sensitive to the values of Δr_{wz} and $\Delta\theta_n$. For a 21×21 grid, $\Delta r_{wz} = 0.02$ and $\Delta\theta_n = 1.0$ gave an error of 0.028053.

Listing of Subroutine GRID1 for Problem 12.11

```
      Subroutine GRID1(JMAX,KMAX,THEB,THEN,RW,RX,RY,RZ)
C
C     MODIFIED VERSION FOR PROBLEM 12.11.
```

```
C
C     SET THE AUGMENTED GRID, INITIAL AND EXACT PHI
C
      DIMENSION XG(23,23),YG(23,23),PHIX(21,21),PHI(21,21)
      DIMENSION THETA(23),RAD(23),RWZ(23)
      COMMON /GRIDP/XG,YG,PHIX,PHI
C
      READ(1,*)DRWZ,DTHN
C
      JMAP = JMAX - 1
      KMAP = KMAX - 1
      AJM = JMAP
      AKM = KMAP
      AJMI = 1./AJM
      AKMI = 1./AKM
      FKK=(KMAX-2)*AKMI
      PI = 3.1415927
      KPP = KMAX + 2
      JPP = JMAX + 2
C
      KMAXX=KMAX+1
      JMAXX=JMAX+1
C
      DO 120 K=2,KMAXX
      AK =(K - 2)
      AK=AK/(KMAXX-2)
      RWZ (K)= RW + (RZ - RW)*AK
      B1 = ((THEN - THEB-DTHN)-(THEN-THEB)*FKK*FKK)
      B1=B1/(FKK*(1.-FKK))
      B2=THEN-THEB-B1
  120 THETA(K) = (THEB + B1*AK + B2*AK*AK)
C
C     THETA FOR REFLECTED CELLS
C
      THETA(1)=THETA(2)-(THETA(3)-THETA(2))
      THETA(KPP)=THETA(KMAXX)+(THETA(KMAXX)-THETA(KMAX))
      RWZ(1)=RWZ(2)
      RWZ(KPP)=RWZ(KMAXX)
C
C
      DO 7 K = 1,KPP
      THK = THETA(K)
      CK = COS(THK*PI/180.)
      SK = SIN(THK*pi/180.)
      RXY = RX + ((THK - THEB)/(THEN-THEB))*(RY - RX)
C
```

```
      DO 6 J = 2,JMAXX
      AJ = J - 2
      AJ=AJ/(JMAXX-2)
      A1=(DRWZ - AJMI*AJMI*(RXY-RWZ(K)))/(AJMI*(1.-AJMI))
      A2= RXY - RWZ(K) -A1
    6 RAD(J) = RWZ(K)+ A1*AJ + A2*AJ*AJ
C
C     RAD FOR REFLECTED CELL
C
      RAD(1)=RAD(2)-(RAD(3)-RAD(2))
      RAD(JPP)=RAD(JMAXX)+(RAD(JMAXX)-RAD(JMAX))
C
C     SET XG, YG, EXACT AND INITIAL PHI
C
      DO 66 J=1,JPP
      XG(J,K) = RAD(J)*CK
      YG(J,K) = RAD(J)*SK
      IF(K .EQ. 1 .OR. K .EQ. KPP)GOTO 66
      IF(J .EQ. 1 .OR. J .EQ. JPP)GOTO 66
      JM = J-1
      KM = K - 1
      PHIX(JM,KM) = SK/RAD(J)
      PHI(JM,KM) = PHIX(JM,KM)
   66 CONTINUE
    7 CONTINUE
      RETURN
       END
```

12.12 For the solution domain (Fig. 12.9) the coordinates in the compu tional and the physical spaces are related by

$\theta = \theta_{WX} + \eta(\theta_{ZY} - \theta_{WX})$,

$r = r_W + \xi(r_X - r_W) + \eta(r_Z - r_W) + \xi\eta[(r_Y - r_Z) - (r_X - r_W)]$,

and $x = r\cos\theta$ and $y = r\sin\theta$.

Now the expressions for the transformation parameters may be derived as : lows:

let $r = r_W + A\xi + B\eta + \xi\eta(C - A)$.

where $A = r_X - r_W$, $B = r_Z - r_W$, $C = r_Y - r_Z$.

From (1) and (4) we have,

$\theta_\xi = 0, \qquad \theta_{\xi\xi} = 0$

$\theta_\eta = \theta_{ZY} - \theta_{WX}, \qquad \theta_{\eta\eta} = 0$

$r_\xi = A + \eta(C - A), \qquad r_{\xi\xi} = 0$

$$r_\eta = B + \xi(C - A), \qquad r_{\eta\eta} = 0 \tag{5}$$

$$x_\xi = r_\xi \cos\theta, \qquad y_\xi = r_\xi \sin\theta\ ,$$

$$x_{\xi\xi} = 0, \qquad y_{\xi\xi} = 0\ .$$

$$x_{\xi\eta} = r_{\xi\eta}\cos\theta - r_\xi\theta_\eta\sin\theta, \qquad y_{\xi\eta} = r_{\xi\eta}\sin\theta + r_\xi\theta_\eta\cos\theta\ ,$$

$$x_\eta = r_\eta\cos\theta - r\theta_\eta\sin\theta, \qquad y_\eta = r_\eta\sin\theta + r\theta_\eta\cos\theta\ ,$$

$$x_{\eta\eta} = -2r_\eta\theta_\eta\sin\theta - r\theta_\eta^2\cos\theta, \qquad y_{\eta\eta} = 2r_\eta\theta_\eta\cos\theta - r\theta_\eta^2\sin\theta\ . \tag{6}$$

The subroutine TRAPA is easily modified to calculate the transformation parameters using (6). The listing of the modified version called TRAPA1 is given below.

Running the program for the test case shown in Table 12.2 the following solution errors are obtained. In the table, error $=\|\theta - \theta_{ex}\|_{rms}$ and N_i is the number of iterations for convergence.

grid	error	N_i
6 × 6	0.07154	15
11 × 11	0.020953	12
21 × 21	0.005285	41

Listing of TRAPA1 for Problem 12.12

```
      Subroutine TRAPA1(JMAX,KMAX,THEB,THEN,RW,RX,RY,RZ)
C
C     FROM GRID COORDINATES CALCULATES TRANSFORM PARAMETERS
C     AS PER THE FORMULA GIVEN IN PROBLEM 12.12.
C
      DIMENSION XG(23,23),YG(23,23),PHIX(21,21),PHI(21,21)
      DIMENSION GWW(21,21),GWT(21,21),GTT(21,21),DELZI(21,21),
     1DELET(21,21)
      COMMON /GRIDP/XG,YG,PHIX,PHI
      COMMON /TRAPP/GWW,GWT,GTT,DELZI,DELET
C
      PI = 3.1415927
      AA=RX-RW
      BB=RZ-RW
      CC=RY-RZ
C
      AJM = JMAX - 1
      AJI = 1. /AJM
      AKM = KMAX - 1
      AKI = 1. /AKM
C
```

```
      DO 2 K = 1,KMAX
      KP = K+1
      AKP=KP - 2
      AKP = AKP * AKI
      KPP = K + 2
      DO 1 J = 1,JMAX
      JP = J + 1
      AJP=JP - 2
      AJP = AJP * AJI
      JPP = J + 2
C
      RAD=SQRT(XG(JP,KP)*XG(JP,KP)+YG(JP,KP)*YG(JP,KP))
      COST=XG(JP,KP)/RAD
      SINT=YG(JP,KP)/RAD
C
      THEE=(THEN - THEB)*PI/180.
      RZETA=AA + AKP*(CC-AA)
      RZE=CC - AA
      RETA=BB + AJP*(CC-AA)
C
C     BASIC TRANSFORM PARAMETERS
C
      XZI = COST*RZETA
      YZI = SINT*RZETA
      XET = COST*RETA - RAD*SINT*THEE
      YET = RETA*SINT + RAD*COST*THEE
      XZZ = 0.0
      YZZ = 0.0
      XEE = -2.*RETA*THEE*SINT - RAD*THEE*THEE*COST
      YEE =  2.*THEE*RETA*COST - RAD*THEE*THEE*SINT
      XZE = COST*RZE - RZETA*SINT*THEE
      YZE = SINT*RZE + RZETA*COST*THEE
      AJ = XZI*YET - XET*YZI
C
C     MODIFIED METRIC TENSOR COEFFICIENTS, G11,G12,G22
C
      GWW(J,K) = (XZI*XZI + YZI*YZI)/AJ
      GWT(J,K) = -2.*(XZI*XET + YZI*YET)/AJ
      GTT(J,K) = (XET*XET + YET*YET)/AJ C
C     MODIFIED DEL**2ZI AND DEL**2ETA
C
      DUM = GTT(J,K)*(XET*YZZ-YET*XZZ)/AJ
      DUM = DUM + GWT(J,K)*(XET*YZE-YET*XZE)/AJ
      DELZI(J,K) = DUM + GWW(J,K)*(XET*YEE-YET*XEE)/AJ
      DUM = GTT(J,K)*(YZI*XZZ - XZI*YZZ)/AJ
      DUM = DUM + GWT(J,K)*(YZI*XZE-XZI*YZE)/AJ
      DELET(J,K) = DUM + GWW(J,K)*(YZI*XEE-XZI*YEE)/AJ
      DELZI(J,K)=DELZI(J,K)/JMAX
      DELET(J,K)=DELET(J,K)/KMAX
    1 CONTINUE
    2 CONTINUE
      RETURN
      END
```

12.13 The modifications to LAGEN to implement the group finite-element method are made in subroutine ITER and the modified version is listed below. The modified program was run for the following conditions : $r_W = r_Z = 0.1$, $r_Y = 1.0$, $\theta_{WX} = 0$, $\theta_{ZY} = 90^0$, $\lambda = 1.5$. The computed solution errors are tabulated below. In the table, error $= \|\theta - \theta_{ex}\|_{rms}$ and N_i is the number of iterations for convergence.

	grid	error	N_i
$r_x = 1$	6×6	0.07514	14
	11×11	0.009701	13
	21×21	0.000971	22
$r_x = 2$	6×6	0.41846	16
	11×11	0.086136	15
	21×21	0.016715	47

Comparing the solution errors now obtained with those in Table 12.2, we find that the finite-element formulation is more accurate. The rate of convergence almost approaches that of a fourth-order method.

But when the mesh gets too distorted (as a consequence of r_x being more than 1), the accuracy reduces considerably.

Listing of ITER1 for Problem 12.13

```
      Subroutine ITER1(JMAX,KMAX,NMAX,NITER,OM,EPS)
C
C     ITERATE USING SOR APPLIED TO DISCRETISED EQUATIONS
C
      DIMENSION GWW(21,21),GWT(21,21),GTT(21,21),DELZI(21,21),
     1DELET(21,21),PHI(21,21),PHIX(21,21),XG(23,23),YG(23,23)
      COMMON /GRIDP/XG,YG,PHIX,PHI
      COMMON /TRAPP/GWW,GWT,GTT,DELZI,DELET
      C1=1./6.
      C2=1.-2.*C1
      KMAP = KMAX-1
      JMAP = JMAX-1
```

```
      AKM = KMAP-1
      AJM = JMAP-1
C
      DO 3 N = 1,NMAX
      SUM = 0.
      DO 2 K = 2,KMAP
      KM = K - 1
      KP = K + 1
      DO 1 J = 2,JMAP
      JM = J - 1
      JP = J + 1
      PHD=-0.5*C1*(DELZI(JP,KM)*PHI(JP,KM)-DELZI(JM,KM)*
     1 PHI(JM,KM))
      PHD=PHD-0.5*C2*(DELZI(JP,K)*PHI(JP,K)-DELZI(JM,K)*
     1 PHI(JM,K))
      PHD=PHD-0.5*C1*(DELZI(JP,KP)*PHI(JP,KP)-DELZI(JM,KP)*
     1 PHI(JM,KP))
C
      PHD=PHD-0.5*C1*(DELET(JM,KP)*PHI(JM,KP)-DELET(JM,KM)*
     1 PHI(JM,KM))
      PHD=PHD-0.5*C2*(DELET(J,KP)*PHI(J,KP)-DELET(J,KM)*
     1 PHI(J,KM))
      PHD=PHD-0.5*C1*(DELET(JP,KP)*PHI(JP,KP)-DELET(JP,KM)*
     1 PHI(JP,KM))
C
      PHD=PHD+C1*(GTT(JM,KM)*PHI(JM,KM)+GTT(JP,KM)*
     1 PHI(JP,KM)-2.*GTT(J,KM)*PHI(J,KM))
      PHD=PHD+C2*(GTT(JM,K)*PHI(JM,K)+GTT(JP,K)*PHI(JP,K))
      PHD=PHD+C1*(GTT(JM,KP)*PHI(JM,KP)+GTT(JP,KP)*
     1 PHI(JP,KP)-2.*GTT(J,KP)*PHI(J,KP))
C
      PHD=PHD+C1*(GWW(JM,KM)*PHI(JM,KM)+GWW(JM,KP)*
     1 PHI(JM,KP)-2.*GWW(JM,K)*PHI(JM,K))
      PHD=PHD+C2*(GWW(J,KM)*PHI(J,KM)+GWW(J,KP)*PHI(J,KP))
      PHD=PHD+C1*(GWW(JP,KM)*PHI(JP,KM)+GWW(JP,KP)*
     1 PHI(JP,KP)-2.*GWW(JP,K)*PHI(JP,K))
C
      PHD=PHD+0.25*(GWT(JP,KP)*PHI(JP,KP)
     1  -GWT(JM,KP)*PHI(JM,KP)
     1 +GWT(JM,KM)*PHI(JM,KM)-GWT(JP,KM)*PHI(JP,KM))
C
      PHD = 0.5*PHD/(C2*GTT(J,K)+C2*GWW(J,K))
      DIF = PHD - PHI(J,K)
      SUM = SUM + DIF*DIF
      PHI(J,K) = PHI(J,K) + OM*DIF
    1 CONTINUE
```

```
  2 CONTINUE
    RMS = SQRT(SUM/AJM/AKM)
    IF(RMS .LT. EPS)THEN
    NITER=N
    RETURN
    ENDIF
  3 CONTINUE
    NITER=N
    WRITE(6,4)NMAX,RMS
  4 FORMAT(' CONVERGENCE NOT ACHIEVED IN'
   1 ,I5,' STEPS,  RMS=',E12.5)
    RETURN
    END
```

CTFD Solutions Manual: Chapter 13

Grid Generation

13.1 This problem is a direct application of (13.7) and (13.80). The solution involves many complex quantities and it will be convenient to use the complex algebra facility available in FORTRAN.

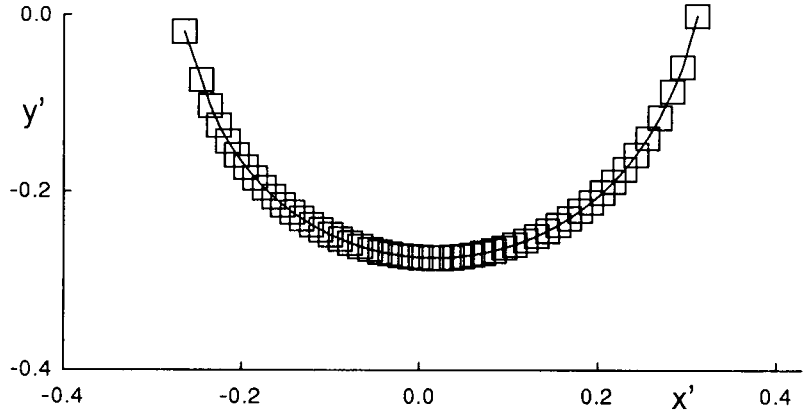

Fig. 1. Points on the near-circle.

The point distribution on the near-circle surface and that on the aerofoil after the inverse transformation are given in Figs. 1 and 2 and in the tables below. The point distribution on the aerofoil preserves its shape properly only when one has a sufficient number of points. The result shown in Fig. 2 is based on 75 points. If too few points are used, the shape may be distorted. The results with a smaller number of points are not shown in the table. Further only every fifth point is provided to allow the reader to check the values.

Coordinates on the Near Circle.

x'	y'	θ
0.311733	-0.000474	-0.08706
0.256719	-0.139348	-28.4932
0.192976	-0.212113	-47.7047
0.131432	-0.250291	-62.2954
0.0714757	-0.269585	-75.1507
0.0130265	-0.274915	-87.2871
-0.0439564	-0.268034	-99.3133
-0.0994675	-0.248846	-111.787
-0.153386	-0.215092	-125.493
-0.205308	-0.159276	-142.196
-0.264115	-0.0188101	-175.926

Points on the Aerofoil Surface.

x	y
0.982545E-06	0.177275E-03
0.147994E-01	0.203927E-01
0.488224E-01	0.349700E-01
0.101192	0.467081E-01
0.170518	0.551881E-01
0.253993	0.594425E-01
0.347994	0.594694E-01
0.448483	0.559887E-01
0.551164	0.497089E-01
0.651691	0.416412E-01
0.745804	0.326113E-01
0.829560	0.236261E-01
0.899430	0.154589E-01
0.952461	0.866073E-02
0.986765	0.474186E-02
1.00000	0.249522E-02

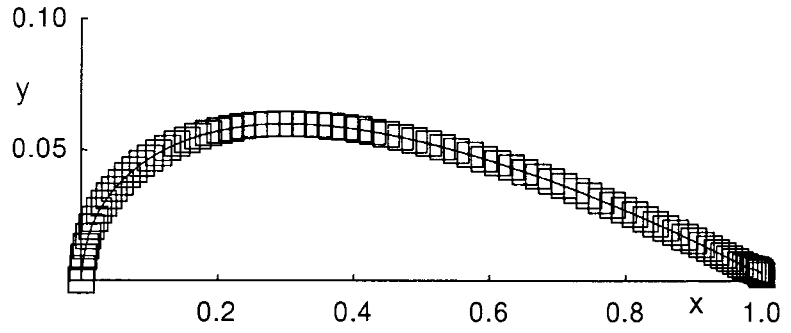

Fig. 2. Grid point distribution on a NACA 0012 aerofoil.

13.2 In the ζ-plane lines of constant ϕ and ψ are prescribed to give a rectangular grid. The x and y coordinates in the physical plane for the various grid points are evaluated from (13.84). The resulting grid (which in fact gives the streamlines and velocity potential lines) for the case when $-10 \leq \phi \leq 10$ and $0 \leq \psi \leq 0.2$, are shown in Fig.1. For an actual computation a much finer grid would be used, particularly close to the corners.

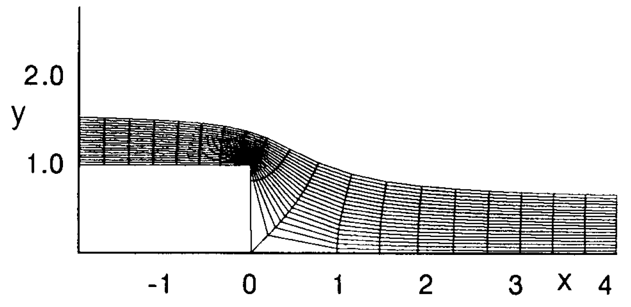

Fig. 1. Streamlines and lines of constant velocity potential for the backward-facing step.

13.3 Modifications to ALGEM to construct an orthogonal grid by the orthogonal trajectory method may be carried out as follows. The first step is to generate the nine intermediate surfaces (see Fig. 13.6 in the text) using subroutine SURCH. Parameters $S_2, S_3, ...$and S_9 are read in and additional arrays $XS4, YS4,.........XS10, YS10$ are set by modifying lines 11 to 18 in SURCH. XS11, YS11 is the outer bounding surface instead of XS4, YS4.

Now it is required to integrate (13.31),

$$\frac{d\mu}{d\nu}\Big|_{\xi=\xi_1} = -\left(\frac{\partial x}{\partial \nu}\frac{\partial x}{\partial \mu} + \frac{\partial y}{\partial \nu}\frac{\partial y}{\partial \mu}\right)\Big/\left[\left(\frac{\partial x}{\partial \mu}\right)^2 + \left(\frac{\partial y}{\partial \mu}\right)^2\right]. \qquad (1)$$

This is a typical characteristic equation and is integrated by marching from ABC to the first intermediate surface and then to the second, until the surface

ED is reached. Equation (1) can be integrated using an algorithm like

$$\mu_k - \mu_{k-1} = RHS \cdot (\nu_k - \nu_{k-1}) \ , \tag{2}$$

where μ_k is the value of μ on the surface $S_k(\nu = \nu_k)$ to be determined. The point (μ_{k-1}, ν_{k-1}) is the corresponding point on surface S_{k-1}. The equivalent coordinates (x_{k-1}, y_{k-1}) are obtained from (13.27).
To improve the grid quality:
1. One can cluster points closer around A in the ξ - direction.
2. The intermediate surface next to ABC should be closer i.e. $\Delta\nu$ should be smaller.
3. Some control is also possible by choosing $EA(\nu)$ and $CD(\nu)$ appropriately in (13.27).

13.4 The procedure to construct a near-orthogonal grid may be described as follows. The method closely follows that of Problem 13.3 and the important change is that now (13.34) is used in place of (13.31).

The procedure described in Sect. 13.2.5 is used to march from one intermediate surface to the other. The method reduces to the determination of the point of intersection of the two normals $(dx/dy)_{\nu_1}$ and $(dx/dy)_{\nu_2}$ with the corresponding $\nu = \nu_2$ lines. The point of intersection of the normal to $\nu = \nu_1$ line with the $\nu = \nu_2$ line is given by the following formulae:

$$x_2 = \left[m_1\{(y_{j1} - y_{j2}) + m_2 x_{j2}\} + x_{j1} \right] \Big/ \left[1 + m_1 m_2\right], \qquad y_2 = y_{j2} + m_2(x_2 - x_{j2}) \ , \tag{1}$$

where m_1 and m_2 are the gradients at j_1 and j_2 of the lines $\nu = \nu_1$ and $\nu = \nu_2$ respectively (see Fig. 13.17). The lines in SURCH implement these formulae. It is necessary to move (x_2, y_2), corresponding to (μ_j, ν_2) in Fig.13.17, until the normal through it passes through (x_{j1}, y_{j1}), corresponding to (μ_j, ν_1).

13.5 The interior grid is to be generated by solving the Poisson equations (13.36a) and (13.36b) by the SOR technique. With reference to Fig. 13.25, the grid points on the lines ABC, CD, DEF and FA must be determined first.

The surfaces ABC and FED are readily determined by ALGEM (lines 64 - 86). The arc lengths along CD and AF are calculated and the grid points are interpolated based on suitable stretching parameters , in a manner equivalent to that used on ABC and FED.

i) Here P=Q=0 in (13.37) and a uniform distribution of points on the boundary is assumed. Note that the functions P and Q control the clustering of the interior points. With P = Q =0, the SOR algorithm for the equations (13.36a) and (13.36b) is given by,

$$\begin{aligned} 2(\alpha' + \gamma')F^*_{j,k} &= \alpha'(F^{n+1}_{j-1,k} + F^n_{j+1,k}) - 0.5\beta'(F^n_{j+1,k+1} - F^{n+1}_{j-1,k+1} \\ &\quad - F^n_{j+1,k-1} - F^{n+1}_{j-1,k-1}) + \gamma'(F^{n+1}_{j,k-1} + F^n_{j,k+1}) = 0 \ , \\ F^{n+1}_{j,k} &= \lambda F^*_{j,k} + (1-\lambda)F^n_{j,k} \ , \end{aligned} \tag{1}$$

where λ is the relaxation parameter and $F = x$ *or* y. For F^n in (1), and for the metric coefficients α', β' and γ', the latest available value of F is used.

The SOR algorithm is implemented by looping over all grid points (j,k). Then the rms value of the change in x and y between the n^{th} and $(n+1)^{th}$ iterations is calculated. The grid is assumed to have converged when the rms change is below a prescribed limit.

ii) The grid determined in the previous section will not be orthogonal. In general, one requires an almost orthogonal grid near the boundaries. To generate such a grid the stretching functions on the boundaries must be chosen so as to cluster points near A along AB, C along CD, A along AF. For an aerofoil computation clustering is also required at B along BA and BC.

In addition to force orthogonality it is necessary to adjust the functions P and Q. This is done by replacing the functions P and Q by P_x and Q_x, P_y and Q_y in (13.36a) and (13.36b) respectively and empirically adjusting them, guided by (13.41) and (13.42).

13.6 Program ALGEM is easily modified to obtain solutions with two (N=2) or three (N=3) multisurfaces.

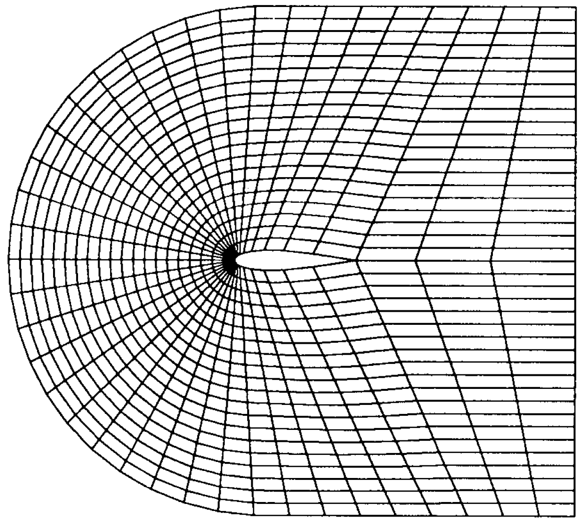
Fig. 1. Grid produced on using two multi surfaces, N=2.

The most important change to the program is to use (13.58) or (13.59) in place of (13.77). In addition subroutine SURCH may be modified to calculate only the S_2 - surface if N=3 and it need not be called if N=2. An integer variable

NSURF is read in. Also changes are made to lines 116 to 127 in ALGEM and are indicated in the listing below.

The grids computed with two and three multisurfaces are shown in Figs. 1 and 2. The quality of the grids is in accordance with the discussion in the text following (13.59). In generating these grids it may be observed that points are clustered around the leading edge. When a limited number of points, say 40 in the ξ direction, are used this produces a less satisfactory grid on the rest of the aerofoil surface and in the wake regions. To produce a grid that can be employed to compute the flow around the aerofoil it is necessary to use a larger number of points, say 100 in the ξ direction.

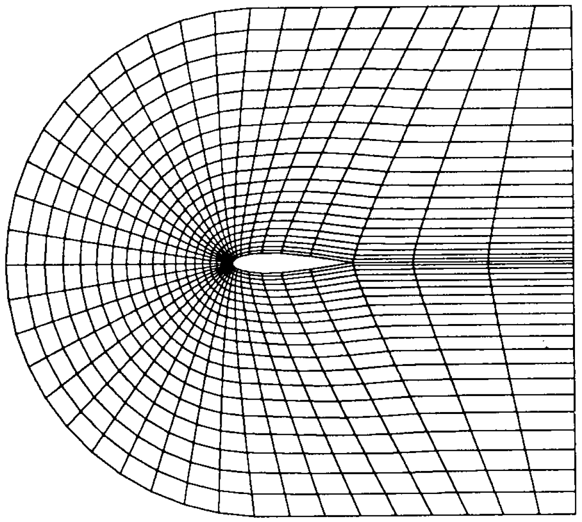
Fig. 2. Grid produced on using three multi surfaces, N=3.

Modifications to ALGEM for Problem 13.6

```
C     FOUR SURFACES
C
      IF(NSURF .EQ. 4) THEN
      SH(1) = (1.-S)**2*(1.-A1*S)
      SH(2) = (1.-S)**2*S*(A1+2.)
      SH(3) = (1.-S)*S*S*(A2+2.)
      SH(4) = S*S*(1.-A2*(1.-S))
      ENDIF
C

C     TWO SURFACES
```

```
C
      IF(NSURF .EQ. 2) THEN
      SH(4)=S
      SH(1)=1.-S
      SH(2)=0.
      SH(3)=0.0
      ENDIF
C
C     THREE SURFACES.
C
      IF(NSURF .EQ. 3) THEN
      SH(1)=(1.-S)*(1.-S)
      SH(2)=2.*S*(1.-S)
      SH(3)=0.0
      SH(4)=S*S
      ENDIF
C
      X(J,K) = 0.
      Y(J,K) = 0.
      DO 22 L = 1,4
      X(J,K) = X(J,K) + SH(L)*XS(L,J)
      Y(J,K) = Y(J,K) + SH(L)*YS(L,J)
   22 CONTINUE
   23 CONTINUE
   24 CONTINUE
```

13.7 This problem requires the determination of XS(1,J), YS(1,J) to follow the surface of the ellipse and XS(4,J), YS(4,J) to follow the outer rectangle shown in Fig.13.24. The following code is inserted after line 49 to achieve this.

Modifications to ALGEM for Problem 13.7

```
C
C   IF INT =2, SET ELLIPTIC SURFACE FOR XS(1, ), YS(1, )
C
      IF(INT .NE. 2) GOTO 104
       RF = 0.5*(XB(1)+XB(2))
       RC=0.98*RF
       AXA = 0.5*(XB(2) - XB(1))
       BYA = AXA/BAR
       DO 100 J = 1,JMAX
       XD = -1.0 + 2.0*RAC(J)
       XS(1,J) = RF + XD*AXA
       DUM = 1.0 - XD**2
       IF(DUM .LT. 1.0E-10) DUM = 1.0 E-10
       YS(1,J) = BYA*SQRT(DUM)
```

```
  100 CONTINUE
C
C     SET OUTER RECTANGULAR BOUNDARY, XS(4, ), YS(4, )
      JMXH = JMAX/2 + 1
      DO 101 JA = 1,JMXH
      J = 2*JA - 1
      ANG = 0.5*PI*RED(J)
      XS(4,JA) = XB(6)
      SANG = SIN(ANG)
      YS(4,JA) = RF*SANG
      IF(YS(4,JA) .LE. YB(5)) GOTO 101
      YS(4,JA) = YB(5)
      CANG = 1.0 - SANG*SANG
      XS(4,JA) = RF*(1.0 - CANG)
  101 CONTINUE
      DO 103 JA = 1,JMXH
      IF((XS(4,JA)-XB(5))**2+(YS(4,JA)-YB(J))**2
     1 .GT. 1.0E-02) GOTO 102
      XS(4,JA) = RF - RC
      YS(4,JA) = 0.98*YB(5)
  102 JB = JMAX - JA + 1
      XS(4,JB) = XB(4) - XS(4,JA)
      YS(4,JB) = YS(4,JA)
      GOTO 175
```

Line 53 becomes statement number 104 and line 91 becomes statement number 175. It is also necessary to modify line 12 to read in INT, BAR and lines 28,29 to set the boundary points to match Fig. 13.24. Running ALGEM with INT=2, BAR=4.0 produces a result equivalent to Fig. 13.24 (c). To obtain results corresponding to N=2,3 requires the modifications discussed in Problem 13.6.

13.8 Modification of program ALGEM to carry out the two-boundary technique (13.49 and 13.50) consists of the following steps.
1. SURCH is not called. The values of T_1 and T_2 are read in.
2. Lines 111 to 124 are modified to make use of (13.49) and (13.50) instead of (13.77). The modified lines are listed later.

The 21 x 21 grid with $T_1 = T_2 = 0.01$ is shown in Fig. 1. The grid is qualitatively good, but would need to be finer for an actual computation. Further, the spacing of points near the outer boundary is fine, a feature which may not be required. A coarser distribution of points in the η - direction near the outer boundary, as in Fig. 13.22, may be obtained by choosing the stretching functions appropriately.

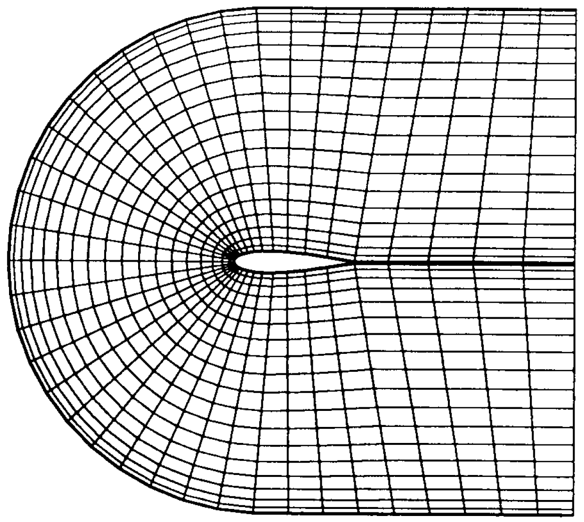
Fig. 1. Grid produced by two-boundary technique.

Modifications to ALGEM for Problem 13.8

```
C     GENERATE INTERIOR GRID
C
      AJM = JMAX - 1
      DZI = 1./AJM
      DO 24 K = 1,KMAX
      DO 23 J = 1,JMAX
      AJ = J - 1
      ZI = AJ*DZI
      S = SAF(K) + ZI*(SCD(K)-SAF(K))
C
      FMU1=2.*S*S*S -3.*S*S +1.
      FMU2=-2.*S*S*S + 3. * S*S
      FMU3=S*S*S -2. *S*S +S
      FMU4=S*S*S - S*S
C
      JPLUS=J+1
      IF(J.EQ.JMAX) JPLUS=J
      JMINUS=J-1
      IF(J.EQ.1) JMINUS=J
      DYAB=(YS(1,JPLUS)-YS(1,JMINUS))/(RAC(JPLUS)-RAC(JMINUS))
      DXAB=(XS(1,JPLUS)-XS(1,JMINUS))/(RAC(JPLUS)-RAC(JMINUS))
```

```
DYDC=(YS(4,JPLUS)-YS(4,JMINUS))/(RFD(JPLUS)-RFD(JMINUS))
DXDC=(XS(4,JPLUS)-XS(4,JMINUS))/(RFD(JPLUS)-RFD(JMINUS))
X(J,K)=FMU1*XS(1,J)+FMU2*XS(4,J)+T1*FMU3*DYAB+T2*FMU4*DYDC
Y(J,K)=FMU1*YS(1,J)+FMU2*YS(4,J)-T1*FMU3*DXAB-T2*FMU4*DXDC
```

13.9 The Vinokur stretching function (J. Comput. Physics, Vol. 50, 1983, pp 215 - 234) is given by

$$s = u/\Big(A + (1-A)u\Big), \tag{1}$$

where $A = (s_0/s_1)^{1/2}$, s_0, s_1 being the slope $d\xi/dt$ at $\xi = 0$ and $\xi = 1$ respectively, and

$$u = 1/2 + \tanh[\Delta y(\xi - 0.5)]/2\tanh(\Delta y/2)\ . \tag{2}$$

Here Δy is the arc length and is typically equal to 1. Of the many forms of equations given by Vinokur , (2) is the one to be used for $B > 1$ where $B = (s_0 s_1)^{1/2}$. Here we calculate s for equally spaced values of ξ.

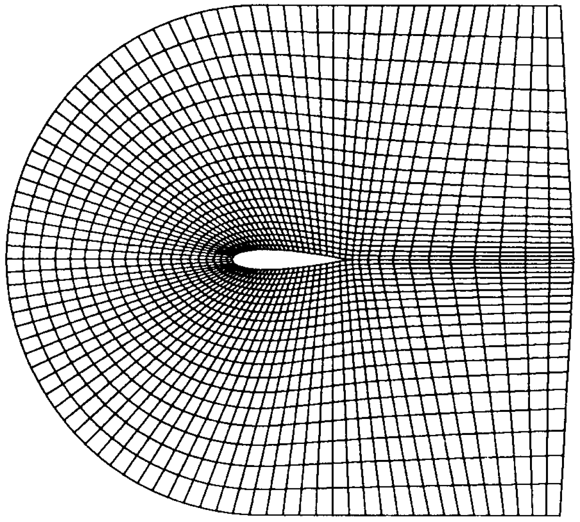
Fig. 1. Grid produced using Vinokur stretching function and N=2.

The listing of the modified form of the subroutine STRECH, called VINOKUR, is given below. To get a reasonably smooth grid the number of points in the ξ direction is about 95. In these calculations, the number of points in the η direction is 21.

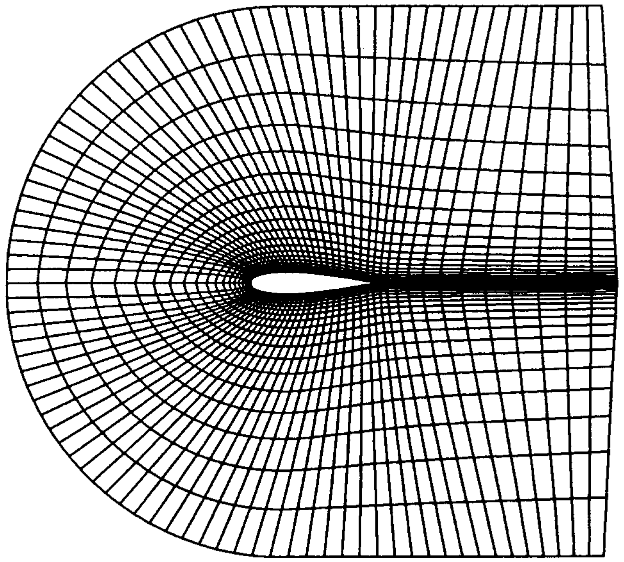
Fig. 2. Grid produced using Vinokur stretching function and N=3.

The generated grid for N=2,3,4 are shown in the following Figs. 1,2 and 3. As pointed out in many references, Vinokur's stretching function is capable of clustering grid points at the ends as well as in the middle. Here an attempt is made to cluster the grid points at A and B (see Fig. 13.25 in the text). To achieve this stretching functions for *AB* and *BC* were generated separately and patched at *B*. This was effected by the following lines of code replacing line 34 in ALGEM. Additional arrays *RABB*, *RBCC*, *RBBB* are used.

```
   JBCC=14
   JABB=JMAX-JBCC
   XB32=XB(3)-XB(2)
   CALL VINOKUR(JABB,25.,5. ,RABB,RAB)
   CALL VINOKUR(JBCC,.5 ,.1 ,RBCC,XB32)
   DO 41 J=2,JBCC
41 RBBB(j-1)=RBCC(j)
   JBCC=JBCC-1
   DO 42 J=1,JBCC
42 RBCC(J)=RBBB(J)
   JMAX=JMAX-1
   DO 43 J=1,JABB
43 RAC(J)=RABB(J)*RAB
   DO 44 J=1,JBCC
44 RAC(JABB+J)=RBCC(J)*XB32+RAC(JABB)
```

```
      RMAX=RAC(JMAX)
      DO 45 J=1,JMAX
   45 RAC(J)=RAC(J)/RMAX
```

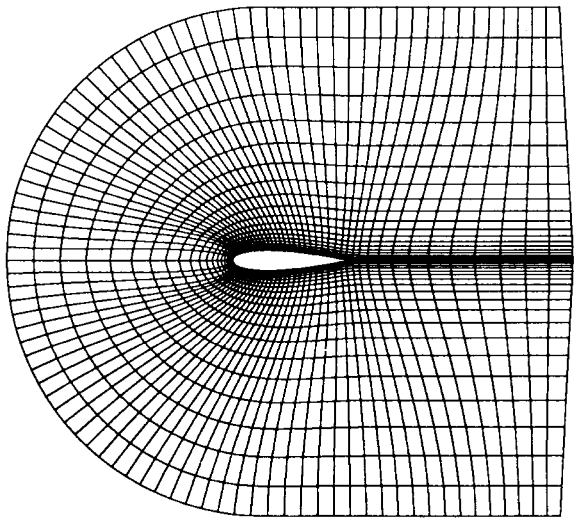
Fig. 3. Grid produced using Vinokur stretching function and N=4.

Modifications to STRECH for Problem 13.9

```
       Subroutine VINOKUR(NX,S0,S1,T,DZ)
       TANH(A)=(EXP(2.*A)-1.)/(EXP(2.*A)+1)
       DIMENSION T(51)
C
C     S0 SLOPE DZI/DS AT ZI=0
C     S1 SLOPE DZI/DS AT ZI=1
C     T IS THE VALUE OF S
C     DZ = DY , TYPICALLY EQUAL TO 1
C
       DZI=1./(NX-1)
       DO 10 I=1,NX
       ZI = DZI*(I-1)
       A=SQRT(S0/S1)
       TANDZ=2.0*TANH(0.5*DZ)
       B=SQRT(S0*S1)
       ARGG=DZ*(ZI-0.5)
```

```
      U=0.5 + TANH(ARGG)/TANDZ
      T(I)=U/(A + (1.-A)*U)
   10 CONTINUE
      RETURN
      END
```

13.10 The equations for transfinite interpolation, to be used for the interior points, are given by (13.66) and (13.67). For the boundary points different procedures are to be used.

i) The two boundary technique, (13.48), is used on the boundaries. This consists of determining the values of x and y for the points on the boundary by formulae equivalent to (13.48), i.e.,

$$x(\xi,0) = (1-r)x_A + rx_C \ , \qquad x(0,\eta) = (1-s)x_A + sx_F \ . \tag{1}$$

With the boundary points determined one can proceed to generate the interior grid. The various steps in the implementation of the transfinite interpolation can be described as follows.

1. Determine the distribution of r along ABC and FED. This can be done with the help of stretching functions as in the previous examples.

2. Calculate for every grid point (J,K) the value of $Z_r(J,K)$ or $Z_r(r,s)$. For this purpose (13.66) is written as

$$Z_r(r,s) = \phi_0(r)Z_{AF}(0,s) + \phi_1(r)Z_{CD}(1,s) \ . \tag{2}$$

If ϕ is chosen as in (13.68) a bilinear interpolation will result.

3. Implement (13.67) as

$$\begin{aligned} Z(r,s) = Z_r(r,s) &+ \psi_0(s)[Z_{ABC}(r,0) - Z_r(r,0)] \\ &+ \psi_1(s)[Z_{FED}(r,1) - Z_r(r,1)] \end{aligned} \ . \tag{3}$$

In the above equations Z is x or y. As indicated in the text, this bilinear interpolation cannot ensure orthogonality of the grid.

ii) In this case we reverse the order in (13.66) and (13.67) to use (13.66) to force orthogonality adjacent to surfaces ABC and FED. Thus the x coordinate of (13.66) is given by

$$\begin{aligned} x_s(r,s) = &\ \mu_1(s)x_{ABC}(r) + \mu_2(s)x_{FED}(r) \\ &+ T_1\mu_3(s)(dy_{ABC}/dr_{ABC})(r) + T_2\mu_4(s)(dy_{FED}/dr_{FED})(r) \ , \end{aligned} \tag{4}$$

and similarly for $y_s(r,s)$ using (13.49) and (13.50). In the second stage the x coordinate of (13.67) is given by

$$\begin{aligned} x(r,s) = x_s(r,s) &+ \phi_0(r)\{x_{AF}(s) - x_s(0,s)\} \\ &+ \phi_1(r)\{x_{CD}(s) - x_s(1,s)\} \ , \end{aligned} \tag{5}$$

where $\phi_0(r)$ and $\phi_1(r)$ are given by (13.68). Some of the modifications used for Problem 13.8 are also useful here.

CTFD Solutions Manual: Chapter 14

Inviscid Flow

14.1 a) The pressure distribution obtained for the flow around a circular cylinder ($B = 1.0$ in program PANEL) with different numbers of panels is shown in Fig. 1. It is evident that the pressure distribution agress well with the theoretical result when the number of panels is equal to or more than 16.

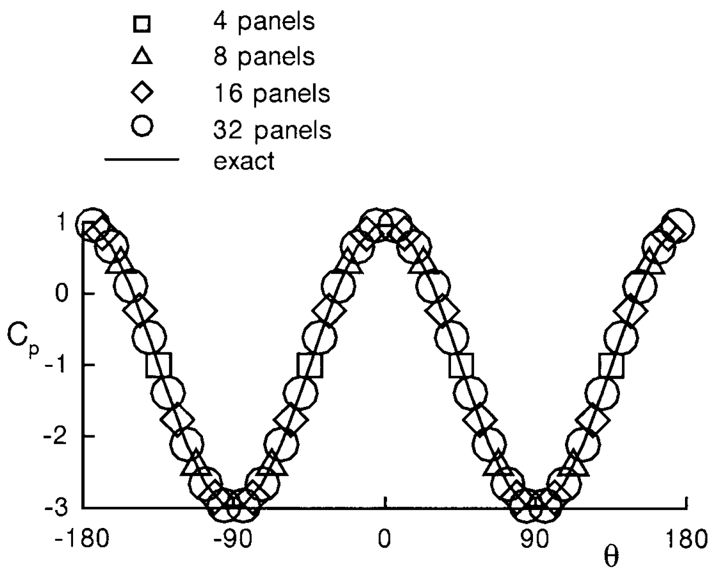

Fig. 1. Pressure distribution for flow around a circular cylinder.

b) The pressure distributions for the flow over an ellipse (with B=0.5 and 0.2) are shown in Figs. 2 and 3 as a function of axial position. Here again, the solution is seen to agree well with theory when the number of panels is 16 or more.

c) It is clear from Figs. 2 and 3 that the accuracy of the panel method increases with the number of panels. Improvement in accuracy for the flow

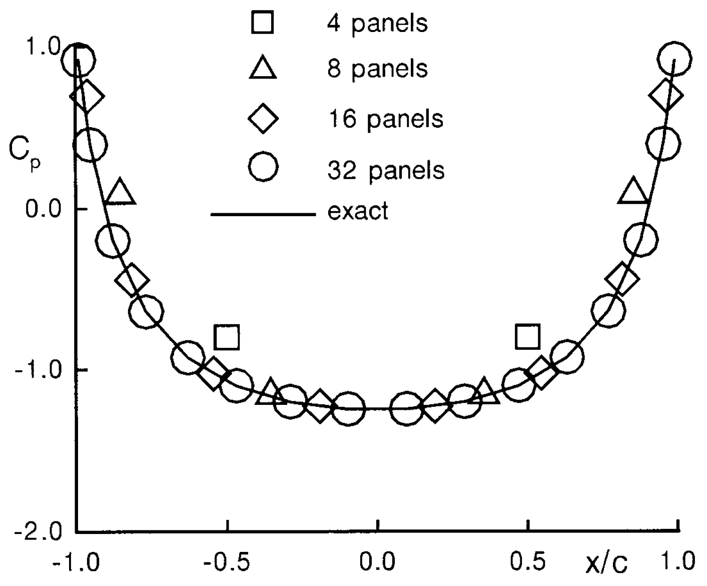

Fig. 2. Pressure distribution for flow around an ellipse, B=0.5.

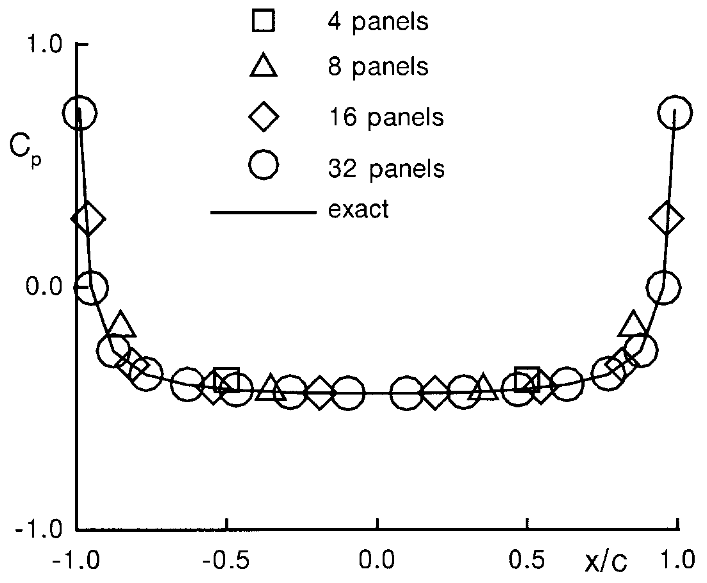

Fig. 3. Pressure distribution for flow around an ellipse, B=0.2.

about an ellipse is also evident as the ratio of *minor/major* axis is reduced (i.e. as B is reduced). However this is mainly due to the smaller disturbance to the flow created by thinner ellipses. For the same number of panels the solution could be improved by clustering more panels at the head and tail of the ellipse where the solution gradients are largest.

14.2 **a)** The C_p distribution computed about a NACA-0012 aerofoil for $M_\infty = 0.4$ at zero incidence with different numbers of panels is shown in Fig. 1.

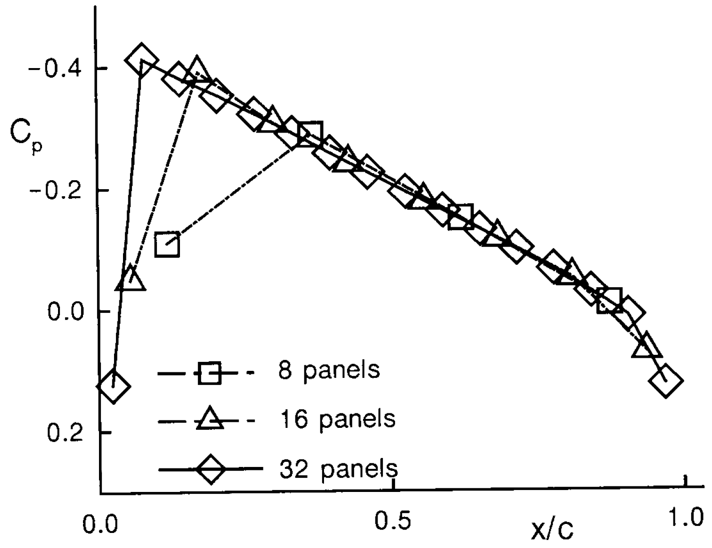

Fig. 1. C_p distribution for flow around a *NACA* aerofoil.

Comparing the present results with those in Fig. 14.4, it is clear that the solutions with 8 and 16 panels are inaccurate near the leading edge. The remedy is to use more panels or to cluster the panels near the leading and trailing edges.

b) The distribution of 16 panels is now based on a cosine rule so that the leading-edge and the trailing-edge regions have more panels than the mid-section. The computed C_p distribution is shown in Fig. 2.
Clearly, there is a substantial improvement in the C_p-prediction near the leading-edge. Still, this distribution varies considerably from that in Fig. 14.4. It is possible to place more panels in the leading edge region to improve the accuracy.

An 'optimal' distribution of panels will mean that the panels are clustered heavily in regions where the flow gradients are high and a coarser distribution is used in regions of uniform or near uniform flow.

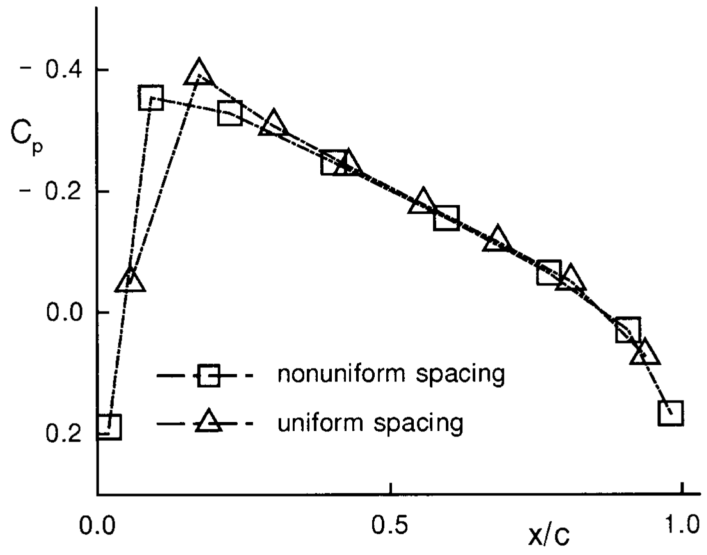

Fig. 2. C_p distribution for flow around a *NACA* aerofoil.

14.3 An *SOR* solver to replace the subroutines FACT and SOLVE is developed based on the algorithm (6.56). Here V is the solution i.e. source density (*SDE*) in Fig. 14.7, A is the matrix AA and B is the right hand side. The listing of the subroutine to implement *SOR* is given later.

Running this version of the PANEL program it was found that the number of iterations to converge was a minimum when $\lambda = 1$. (λ is the relaxation factor, see Sect. 6.3.2). The number of iterations required for convergence for the flow about a circle and about a NACA 0012 (in brackets) aerofoil are given below. In the table NP denotes the number of panels.

NP	$\lambda = 1.0$	$\lambda = 1.2$	$\lambda = 1.5$
8	12 (53)	19 (43)	37 (26)
16	13 (53)	19 (33)	70 (26)
32	13 (50)	19 (32)	76 (26)
48	13 (49)	20 (31)	100 (26)

Thus, we see that the number of iterations to converge does not seem to depend upon the number of panels used, except when the number of panels is high. This character may be specific to the present application.

Modifications to PANEL for Problem 14.3

```
      Subroutine SOR(V,B,A,N,FLAM,ITMAX,TOL)
      DIMENSION V(50),B(50),A(50,50)
C     FLAM, ITMAX, TOL ARE READ IN THE MAIN PROGRAM.
      AN=N
      FLAMM=1. - FLAM
      NITER = 0
    5 NITER = NITER +1
      SUMR = 0.0
      DO 20 I=1,N
      DUM = B(I)
      DO 10 J= 1,N
      IF(I.EQ.J) GO TO 10
   10 DUM = DUM - A(I,J)*V(J)
      RES = DUM - A(I,I)*V(I)
      DUM=DUM/A(I,I)
      DUM = FLAM * DUM + FLAMM * V(I)
      DIFF= DUM - V(I)
      V(I)=DUM
   20 SUMR = SUMR + RES*RES/AN
      RMSR=SQRT(SUMR)
      IF(NITER.GT.ITMAX) GO TO  30
      IF(RMSR . GT. TOL) GO TO 5
      WRITE(*,*) ' CONVERGENCE OBTAINED'
      WRITE(*,*)NITER
      RETURN
   30 WRITE(*,*)' NO CONVERGENCE'
      WRITE(*,*)NITER
```

14.4 a) The operation count for the various subroutines in PANEL, taking into account only multiplication and division, are as follows. Each mathematical library function (Table 4.4) used in MATELM is assumed equal to twenty multiplications.

Subroutine	Operation count for for N panels	Operation count for for 22 panels
BODY	$6N + 3N/2 + 3$	168
MATELM	$202N^2 + 3N$	97834
SURVL	$2N^2 + 21N + 8$	1438
POINT	$21N + 7$	470
FACT	$N^3/3 + 0.6N^2$	3840
SOLVE	N^2	484

If the number of panels is 22 the total operation count is 104234. For small values of N, subroutine MATELM dominates the operation count due to the relatively expensive evaluation of the mathematical functions. However for N sufficiently large, subroutine FACT will eventually be the most expensive part of the program to evaluate.

b) When an SOR solver is used in place of FACT and SOLVE it requires $N^2 + 4N$ operations per iteration. It required 9 iterations to converge when $N = 22$.

The total operation count when SOR is used is: 105058

For small values of N the iterative solver provides no advantage, particularly since the operation count is dominated by subroutine MATELM. As N increases the number of iterations to convergence is expected to increase slowly. Eventually, for sufficiently large N, it is more economical to obtain the solution using an iterative solver.

Program LAGEN (with an $N \times N$ grid) requires the following number of operations. In the table N_i denotes the number of iterations to converge.

Subroutine	Operation count for for N panels	Operation count for for 22 panels
GRID	$4N^2 + 47N + 3$	2973
TRAPA	$46N^2$	22264
ITER	$19N^2N_i$	$9196N_i$

From Table 12.2, we see that for $r_x = 1.0$ and a 21×21 grid, LAGEN requires 53 iterations.

Hence the operation count is: 512,000

The program LAGEN requires more operations than PANEL because the former has N^2 grid points. It may be recalled that PANEL has only N grid points, effectively. If subroutine MATELM were made more efficient by avoiding some of the mathematical function evaluations the contrast would be even greater.

14.5 Modification of program PANEL to compute solutions about lifting aerofoils is carried out using (14.22) and (14.24). The procedure may be outlined as follows. First a source strength distribution σ for a non-lifting flow is calculated making use of the freestream velocity components U_∞ and V_∞. Then a unit strength vortex is placed at the centre of every panel. This produces a lifting solution, but the boundary condition of zero normal velocity at the control points is not met. To counteract this component of velocity, auxiliary sources of strength σ^c are prescribed. As described in Sect.14.1.4 imposition of the Kutta condition gives the weighting τ to be applied to the solution with the vortex and the auxiliary source. The formula (14.24) is modified to include the effect of the vortex distribution, thus,

$$\tau = \frac{-\left[\sum_j \left(FT_{l,j} + FT_{m,j} + U_\infty(\cos\alpha_l + \cos\alpha_m) + V_\infty(\sin\alpha_l + \sin\alpha_m)\right)\right]}{\sum_j \left(FT_{l,j} + FT_{m,j}\right)\sigma_j^c + \sum_j \left(FN_{l,j} + FN_{m,j}\right)} .$$

By combining the effects of the sources of strength σ, vortices of strength τ, auxiliary sources of strength of σ^c and the free stream components the lifting solution about an aerofoil is obtained. The caculated pressure field is integrated to obtain the lift coefficient C_L. Modifications to program PANEL to compute the lifting solution are shown in the listing given later.

The modified program was run for a NACA-0012 aerofoil for M_∞=0, and α= 0, 2 and 4^0 incidence. Forty eight panels were uniformly placed around the aerofoil. The computed C_p distribution for various angles of incidence are shown in Fig. 1 and the computed values of C_L and the theoretical values given by $C_L = 2\pi\alpha$ are given in the table. It is observed that the computed value is about 10% higher than the theoretical. It is to be noted that the theoretical formula is strictly valid for a flat plate.

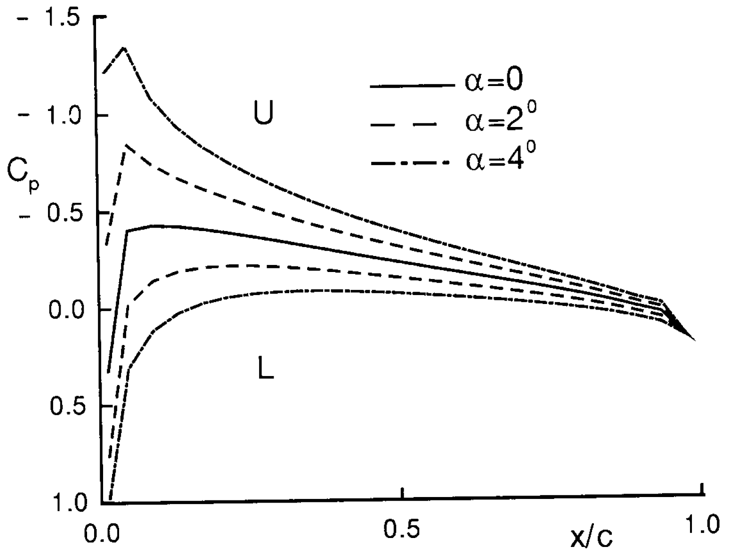

Fig. 1. C_p distribution for lifting flow around a *NACA* aerofoil.U and L denote upper and lower surfaces respectively.

α	C_L, computed	C_L, theory
0	0.0	0.0
2^0	0.2442	0.2193
4^0	0.4873	0.4387

Modifications to PANEL for Problem 14.5

Extra arrays $RHSC$ and $SDEC$ are employed and each has the same dimension as RHS and SDE. Line 12 is modified to read in α (called ATCK in the code) and it is converted to radians after line 23. The call to MATELM (line 32) is modified as

```
      CALL MATELM(N,M,IPR,RHSC)
```

The first line in MATELM is changed accordingly and $RHSC$ is added to the DIMENSION statement. The following lines are inserted after line 47 in PANEL.

```
      DO 200 K=1,N
      SDEC(K)=RHSC(K)
  200 CONTINUE
      CALL SOLVE(N,AA,IKS1,SDEC)
C
      NHLFF=N/2 + 1
      NL=NHLFF-1
      NM=NHLFF
C
      FTSD=0.0
      FTSDC=0.0
      FVU=0.0
      DO 300 J=1,N
      FTSD=FTSD + (FT(NL,J)+ FT(NM,J))*SDE(J)
      FTSDC=FTSDC+(FT(NL,J)+FT(NM,J))*SDEC(J)
     1  + FN(NL,J) + FN(NM,J)
  300 CONTINUE
C
      TOW=-(FTSD + UINF*(CI(NL)+CI(NM))
     1  + VINF*(SI(NL)+SI(NM)))/FTSDC
      DO 301 K=1,N
  301 SDEC(K)= TOW*SDEC(K)
C
      CALL SURVL(N,B,FMN,TOW)
      CLTH=2.*PI*ATCK
      WRITE(6,*)'THEORITICAL CL>',CLTH
```

In subroutine MATELM, $RHSC$ is added to the DIMENSION statement and the line $RHSC(K) = 0$ is introduced after line 10 . The following statement is introduced after line 35.

```
         RHSC(K)=RHSC(K)+FT(K,J)
```

The first line in sunroutine SURVL is changed adding TOW to the list of arguments and $SDEC$ is added to the DIMENSION statement. Lines 17 to 41 are replaced by:

```
      CL=0.0
      DO 4 K=1,N
      QTS=0.0
      QNS=0.0
      DO 2 J=1,N
      QTS=QTS+FT(K,J)*SDE(J)
      QNS=QNS+FN(K,J)*SDE(J)
      QTS=QTS+FT(K,J)*SDEC(J)+FN(K,J)*TOW
      QNS=QNS+FN(K,J)*SDEC(J)-FT(K,J)*TOW
    2 CONTINUE
C
      QNK = QNS + VINF*CI(K) - UINF*SI(K)
      QTK = QTS + VINF*SI(K) + UINF*CI(K)
C
      UU=UINF-QNS*SI(K)+QTS*CI(K)
      VV=VINF+QNS*CI(K)+QTS*SI(K)
      UU = UU/FAC/FAC
      VV = VV/FAC
      PP=1.-UU*UU-VV*VV
      IF(FMN .GT. 0.05)PP = ((1.+C1*PP)**GMP-1.)/C2
C
C     CALCULATION OF CL.
C
      XX=X(K+1)-X(K)
      CL=CL+ PP *XX
      WRITE(6,3)XC(K),YC(K),QNK,QTK,UU,VV,PP,QEX
    3 FORMAT(1X,'XC,YC=',2F6.3,'  QN,QT=',2F6.3,
     1       '  U,V=',2F6.3,'  P=',F6.3,'  QEX=',F6.3)
    4 CONTINUE
      WRITE(6,*)'CL>',CL
```

14.6 Application of program SHOCK with the MacCormack scheme ($IME = 1$) to the propagating shock problem (pressure ratio = 2.5) produces the shock profile shown in the Fig.1. The program is applied with and without artificial viscosity. The profile given by the Lax-Wendroff scheme is also shown for comparison.

The MacCormack scheme gives a slightly lower overshoot behind the shock than the Lax-Wendroff scheme, but the profiles are similar. With artificial viscosity the two schemes give almost identical profiles. This is due to the fact that the shock profile is controlled more by artificial viscosity than by the basic scheme.

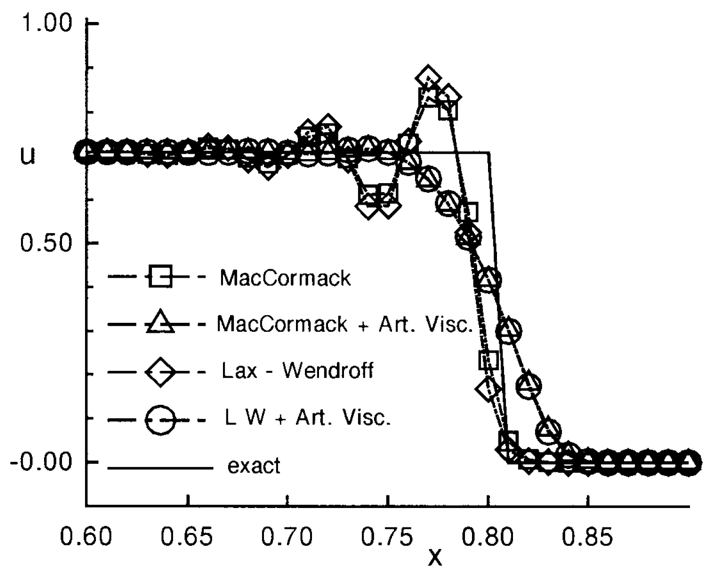

Fig. 1. Shock profiles given by various schemes.

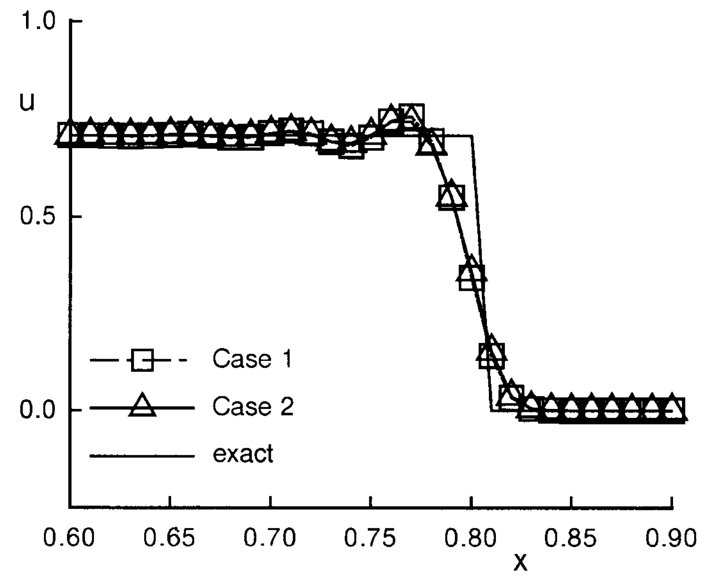

Fig. 1. (Problem 14.7) Shock profile given by Lax-Wendroff scheme with the Lapidus form of artificial viscosity, $p_1/p_2 = 2.5$.

14.7 The form of artificial viscosity due to Lapidus (14.168) is applied with the Lax-Wendroff scheme for the propagating shock. As case 1) the artificial viscosity is applied to all the components of q and as case 2) it is applied only to the first and third components of q.

The computed velocity profiles are shown in Figs. 1 and 2. In comparison with the form of artificial viscosity given by (14.53), the Lapidus form gives a sharper shock profile but a post-shock oscillation is present. Whether the artificial viscosity is applied to all the components of q (case 1) or to only the first and the third ones (case 2) , does not seem to influence the shock profile.

Conclusions are similar for the strong shock example (i.e. pressure ratio =5).

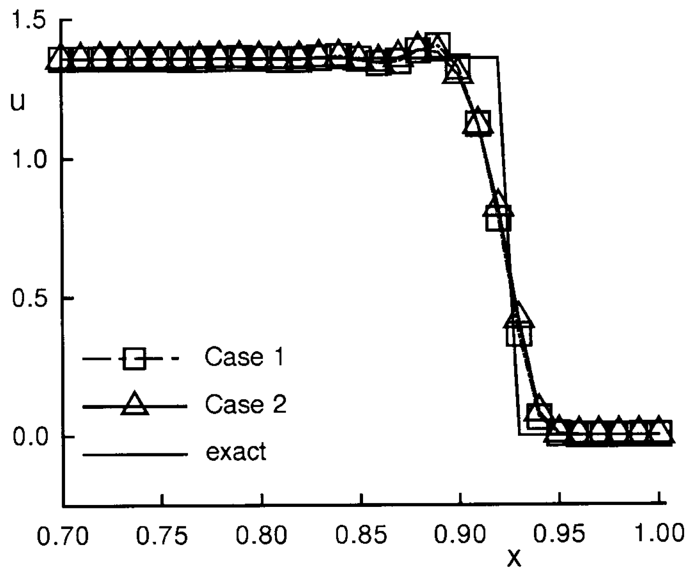

Fig. 2. (Problem 14.7) Shock profile given by Lax-Wendroff scheme with the Lapidus form of artificial viscosity, p_1/p_2 =5.

14.8 Program SHOCK is modified to compute the flow in a shock tube. The major change in the program is in the imposition of the starting conditions. The program SHOCK calculates the post-shock conditions and imposes them as starting conditions. Now these are replaced by u'_1, ρ'_1 and p'_1 being the conditions upstream of the diaphragm before it is burst, i.e. at $t = 0$.

The density and pressure profiles given by the Lax-Wendroff scheme with artificial viscosity are shown in Fig. 1. The shock and the contact discontinuity (seen as a severe density gradient between the shock and the expansion wave) profiles have oscillations upstream of them. The contact discontinuity profile is seen to be greatly smeared.

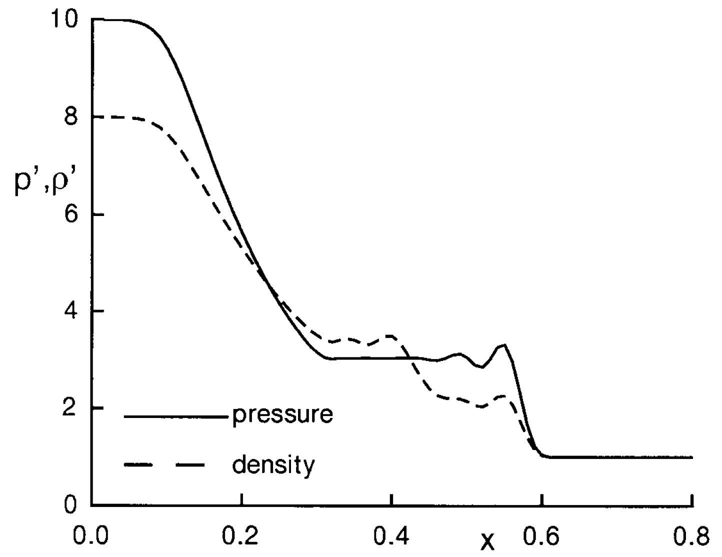

Fig. 1. Shock tube flow predicted by Lax-Wendroff scheme.

The exact solution to this flow is calculated using the equations given by Liepmann and Roshko (see problem definition for details) and is shown in the following table. Pressure and density values given in the table have been normalised with respect to those downstream of the shock.

Region	pressure	density
1) between shock and contact discontinuity	3.03	2.124
2) between contact discontinuity and foot of the expansion wave	3.03	3.409

The computed results agree closely with these values. The density profile on either side of the contact discontinuity is oscillatory, but a mean curve would give the values tabulated above.

14.9 The shock profiles computed with the *FCT* scheme are given in Fig. 1 ($p_1/p_2 = 2.5$) and Fig. 2 ($p_1/p_2 = 5$).

The velocity profiles for the moderate shock strength is free of oscillations except that a small kink develops for the case i) which gives a sharp shock profile spread within only two grid points. The other cases give a 'three-point' shock.

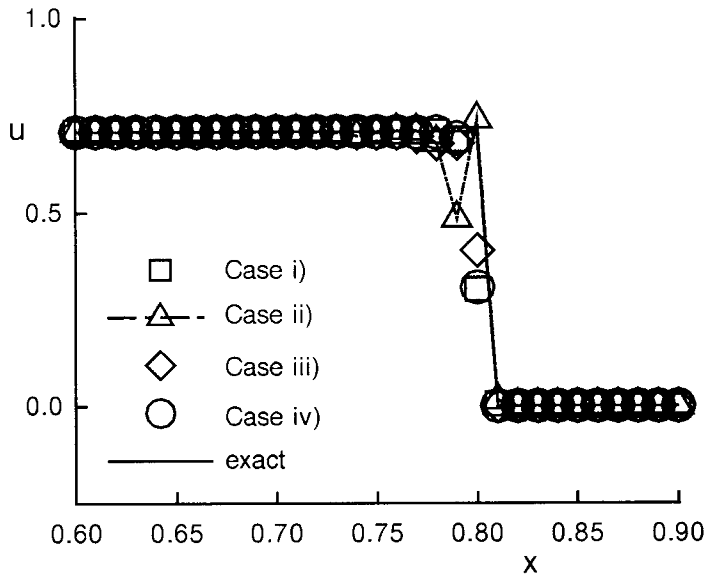

Fig. 1. Shock profile given by FCT algorithm, $p_1/p_2 = 2.5$.

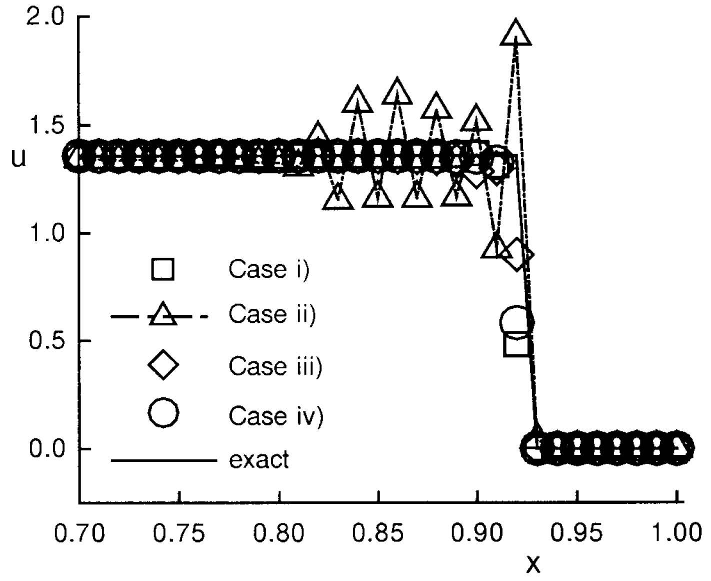

Fig. 2. Shock profile given by FCT algorithm, $p_1/p_2 = 5.0$.

But, for the strong shock, we notice that case ii) gives an unacceptable profile. The other cases produce a 'three-point' shock as before. Case iii) is slightly too dissipative. Cases i) and iv) are comparable except that case i) has a slight trailing oscillation. By varying the values of η_o, η_1 and η_2, one may get a clean shock profile even for a strong shock.

14.10 Applying the FCT algorithm for the shock-tube flow, the results shown in Fig. 1 are obtained.

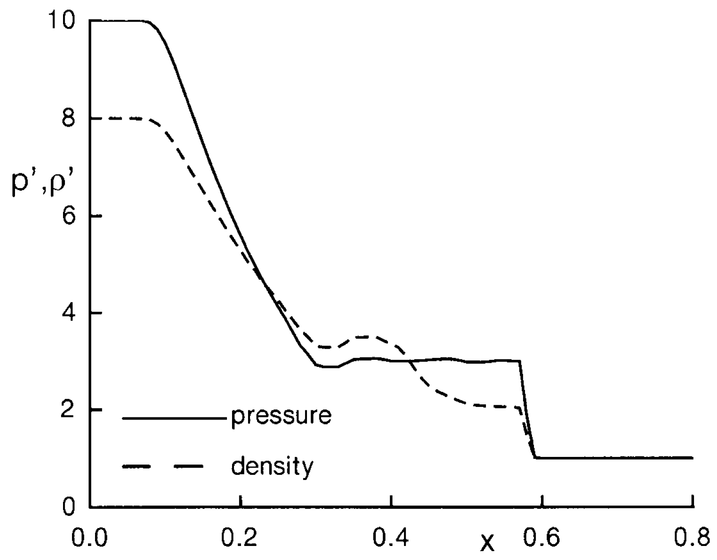

Fig. 1. Shock tube flow predicted by FCT algorithm.

Comparing with the results for Problem 14.8, we notice a sharper shock profile and a cleaner pressure prediction upstream of the shock. The contact discontinuity behaviour shows no improvement. Adjusting the values of η_o, η_1 and η_2 in the scheme will help to improve the performance of the scheme near the contact discontinuity. FCT algorithms are generally more effective in producing sharp shock profiles than in producing sharp profiles across contact discontinuities.

14.11 The various steps involved in the use of SOR iteration to compute the transonic, small disturbance equation for the domain shown in Fig. 14.34 are as follows:

1. Construct an appropriate grid. The two boundary technique described in Sect. 13.3.2 can be used. The lower boundary is defined by $y = 0$ for $x \leq -1$ and $x \geq 1$, and $y = \tau(x^2 - 1)^2$ for $-1 \leq x \leq 1$. The top boundary is placed at $y = 1$. The boundaries AD and BC are placed at $x = -2$ and $x = 2$. A suitable number of points may be chosen in the x and y directions, say 41×21.

2. Develop an SOR algorithm to solve (14.135) i.e.

$$P_{j,k} + Q_{j,k} - \mu_{j,k}P_{j,k} + \mu_{j-1,k}P_{j-1,k} = 0 \ . \tag{1}$$

Substituting the expressions for P and Q, given by (14.134), the above equation simplifies to

$$\begin{aligned}
&(1-\mu_{j,k})\bigg(\frac{k}{\Delta x^2}(\phi_{j+1,k} - 2\phi_{j,k} + \phi_{j-1,k}) - \\
&\frac{0.5(\gamma+1)}{\Delta x^2}(\phi^2_{j+1,k} - \phi^2_{j-1,k} + 2\phi_{j,k}(\phi_{j-1,k} - \phi_{j+1,k}))\bigg) \\
&+ \mu_{j-1,k}\bigg(\frac{k}{\Delta x^2}(\phi_{j,k} - 2\phi_{j-1,k} + \phi_{j-2,k}) - \\
&\frac{0.5(\gamma+1)}{\Delta x^2}(\phi^2_{j,k} - \phi^2_{j-2,k} + 2\phi_{j-1,k}(\phi_{j-2,k} - \phi_{j,k}))\bigg) \\
&+ \frac{\phi_{j,k+1} - 2\phi_{j,k} - \phi_{j,k-1}}{\Delta y^2} = 0 \ .
\end{aligned} \tag{2}$$

Let $k/\Delta x^2 = C_1, \quad 0.5(\gamma+1)/\Delta x^2 = C_2, \quad 1/\Delta y^2 = C_3$.

The above equation can be rearranged as

$$\begin{aligned}
&\phi_{j,k}\bigg(-2(1-\mu_{j,k})C_1 - 2C_2\phi_{j-1,k} + 2C_2\phi_{j+1,k} + \mu_{j-1,k}C_1 \\
&- \mu_{j-1,k}C_2\phi_{j,k} + \mu_{j-1,k}C_2\phi_{j-1,k} - 2C_3\bigg) \\
&= -\Big((1-\mu_{j,k})\{C_1(\phi_{j+1,k} + \phi_{j-1,k}) - C_2(\phi^2_{j+1,k} - \phi^2_{j-1,k})\}\Big) \\
&- \mu_{j-1,k}\Big(C_1(-2\phi_{j-1,k} + \phi_{j-2,k}) - C_2(-\phi^2_{j-2,k} + 2\phi_{j-1,k}\phi_{j-2,k})\Big) \\
&- C_3\Big(\phi_{j,k+1} + \phi_{j,k-1}\Big) \ .
\end{aligned} \tag{3}$$

Equation (3) can be solved using the SOR technique described in Sect. 6.3.

NOTE: a) Typically, using SOR, the solution at the $(n+1)^{th}$ iteration level is given by $\quad \phi^{(n+1)}_{j,k} = \lambda\phi^{(*)}_{j,k} + (1-\lambda)\phi^{(n)}_{j,k}$. (4)

While using (3), the $\phi_{j,k}$ terms appearing within the brackets on the LHS, are replaced by $\phi^{(n)}_{j,k}$.

b) The terms within the brackets on the LHS, may be small at times, leading to possible instabilities. Special techniques may have to be used, such as limiting the smallest size of the bracketed terms.

3. The iteration is carried out from $I = 2$ to $NX - 1$ and from $J = 2$ to $NY - 1$. The values of ϕ on the boundaries are supplied through the boundary conditions.

14.12 The pseudo-transient equivalent of (14.137) can be written as

$$\frac{\partial\phi}{\partial t} + \frac{\partial}{\partial x}\left(\rho'\frac{\partial\phi}{\partial x}\right) + \frac{\partial}{\partial y}\left(\rho'\frac{\partial\phi}{\partial y}\right) = 0 \ , \tag{1}$$

and density is governed by

$$\rho' = \left\{1 + 0.5(\gamma-1)M^2_\infty\left[1 - \left(\frac{u}{U_\infty}\right)^2 - \left(\frac{v}{U_\infty}\right)^2\right]\right\}^{1/(\gamma-1)} \ . \tag{2}$$

Equation (1) is discretised as,

$$\frac{\partial\phi}{\partial t} + S_{j,k} = 0 \ , \tag{3}$$

$$\begin{aligned}
\text{where} \quad S_{j,k} = &[(\tilde\rho\frac{\partial\phi}{\partial x})_{j+1/2,k} - (\tilde\rho\frac{\partial\phi}{\partial x})_{j-1/2,k}]\frac{1}{\Delta x} \\
&+ [(\rho'\frac{\partial\phi}{\partial y})_{j,k+1/2} - (\rho'\frac{\partial\phi}{\partial y})_{j,k-1/2}]\frac{1}{\Delta y} \ .
\end{aligned} \tag{4}$$

$\tilde\rho$ is the modified density term (see 14.143) to introduce artificial viscosity

and $\tilde\rho_{j+1/2,k} = (1-\nu_{j,k})\rho_{j+1/2,k} + \nu_{j,k}\rho_{j-1/2,k}$. (5)

The definition of $\nu_{j,k}$ etc. can be obtained from the text.

It may be noted that density terms are not replaced in the y-derivatives. The given body is slender and any supersonic flow is expected to be only in the x-direction.

Now, substituting for $\tilde\rho$ and terms like $(\partial\phi/\partial x)_{j+1/2,k}$

i.e. $\quad (\partial\phi/\partial x)_{j+1/2,k} = (\phi_{j+1,k} - \phi_{j,k})/\Delta x$, (4) becomes,

$$\begin{aligned}
&\frac{\partial\phi}{\partial t} + \frac{1}{\Delta x^2}\Big[\big((1-\nu_{j,k})\rho_{j+1/2,k} + \nu_{j,k}\rho_{j-1/2,k}\big)(\phi_{j+1,k} - \phi_{j,k}) \\
&\quad - \Big((1-\nu_{j-1,k})\rho_{j-1/2,k} + \nu_{j-1,k}\rho_{j-3/2,k}\Big)(\phi_{j,k} - \phi_{j-1,k})\Big] \\
&\quad + \frac{1}{\Delta y^2}\Big[\rho_{j,k+1/2}(\phi_{j,k+1} - \phi_{j,k}) - \rho_{j,k-1/2}(\phi_{j,k} - \phi_{j,k-1})\Big] = 0 \ .
\end{aligned} \tag{6}$$

Equation (6) can be written in terms of corrections as,

$$\Delta\phi^{n+1} + \frac{\Delta t}{\Delta x^2}(\qquad)^{n+1} + \frac{\Delta t}{\Delta y^2}(\qquad)^{n+1} = 0 \ . \tag{7}$$

Carrying out the linearisation of ϕ terms, the above equation simplifies to

$$1-\frac{\Delta t}{\Delta x^2}\Big\{\left[(1-\nu_{j,k})\rho_{j+1/2,k}+\nu_{j,k}\rho_{j-1/2.k}\right](\Delta\phi_{j+1,k}-\Delta\phi_{j,k}))$$
$$-\left[(1-\nu_{j-1,k})\rho_{j-1/2,k}+\nu_{j-1,k}\rho_{j-3/2,k}\right](\Delta\phi_{j,k}-\Delta\phi_{j,k-1})\Big\} \qquad (8)$$
$$-\frac{\Delta t}{\Delta y^2}\left[\rho_{j,k+1/2}(\Delta\phi_{j,k+1}-\Delta\phi_{j,k})-\rho_{j,k-1/2}(\Delta\phi_{j,k}-\Delta\phi_{j,k-1})\right]$$
$$=-(RHS)^n\ ,$$

where $(RHS)^n$ is the sum of all the spatial derivatives in (6) evaluated at the 'n' level.

Equation (8) is of the form $N\Delta\phi^{n+1}=-\omega R$, (9)

where ω is a scaling factor and $R=(RHS)^n$.

Following (14.156) and (14.157) the approximate factorisation technique is implemented as

$$\{1-\Delta t L_x^+(\tilde\rho)L_x^-\}\,\Delta\phi^*_{j,k}=\omega R_{j,k}\Delta t\ , \qquad (10)$$

and $\{1-\Delta t L_y^+(\rho')L_y^-\}\,\Delta\phi_{j,k}=\Delta\phi^*_{j,k}$. (11)

Now $\left(L_x^+(\tilde\rho)L_x^-\right)\Delta\phi^*_{j,k}$

$$=\frac{1}{\Delta x^2}\Big\{\left[(1-\nu_{j,k})\rho_{j+1/2,k}+\nu_{j,k}\rho_{j-1/2,k}\right](\Delta\phi_{j+1,k}-\Delta\phi_{j,k}) \qquad (12)$$
$$-\left[(1-\nu_{j-1,k})\rho_{j-1/2,k}+\nu_{j-1,k}\rho_{j-3/2,k}\right](\Delta\phi_{j,k}-\Delta\phi_{j-1,k})\Big\}\ ,$$

$$\left(L_y^+(\rho')L_y^-\right)\Delta\phi^{n+1}_{j,k} \qquad (13)$$
$$=\frac{1}{\Delta y^2}\left[\rho_{j,k+1/2}(\Delta\phi_{j,k+1}-\Delta\phi_{j,k})-\rho_{j,k-1/2}(\Delta\phi_{j,k}-\Delta\phi_{j,k-1})\right]\ .$$

The various steps involved in the implementation of this technique are as follows.

1. Construct a suitable grid similar to the one generated for Problem 14.11.
2. Impose the starting values at every grid point.
3. Choose a 'time' step Δt.
4. Calculate the residual R at every grid point.
5. Perform the x-direction sweeps using (12) and calculate $\Delta\phi^*$ at every grid point.
6. Using (13) perform y-direction sweeps and calculate $\Delta\phi^{n+1}$.
7. Update the ϕ values and calculate ρ at every grid point.

8. Repeat steps 4-7 until the rms value of RHS is below a prescribed tolerance.

14.13 i) When we apply an equivalent approximate factorisation to (14.156, 14.157) to the Laplace equation, $\phi_{xx}+\phi_{yy}=0$, we have

$$\{1-\alpha L_{xx}\}\Delta\phi^*_{j,k}=\alpha\omega R_{j,k}\ , \qquad (1)$$

$$\{1-\alpha L_{yy}\}\Delta\phi_{j,k}=\Delta\phi^*_{j,k}\ . \qquad (2)$$

$R_{j,k}$ is the RHS given by $\partial^2\phi/\partial x^2+\partial^2\phi/\partial y^2 = L_{xx}\phi+L_{yy}\phi$. (3)

Using central differences, $L_{xx}\phi=(\phi_{j+1}-2\phi_j+\phi_{j-1})/\Delta x^2$ etc. (4)

It is straightforward to discretise (1) and (2). A program developed to solve (1) and (2) is listed below.

The errors obtained on running this program for various grids are shown below.

Grid	No. of iterations	Rms Error of Corrections	Rms$_{RHS}$	Solution Error
6×6	64	0.77×10^{-6}	0.127×10^{-7}	0.256×10^{-2}
11×11	62	0.97×10^{-6}	0.159×10^{-7}	0.650×10^{-3}
21×21	81	0.86×10^{-6}	0.960×10^{-10}	0.166×10^{-3}

Listing of code for Problem 14.13

```
      Program PR14P13
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION PHI(65,65),PHIO(65,65),A(5,65),R(65)
     1,DUM(65),XX(65),YY(65),PHIEX(65,65),CORR(65,65)
C
C     READ CONTROL PARAMETERS.
C
      OPEN(5,FILE='PR1413.DAT')
      REWIND 5
      READ(5,*)NX,NY,ALPHA,WOM,NLIMIT,RLIMIT
      OPEN(8,FILE='PR1413.OUT')
C
      NXX=NX-1
      NX2=NX-2
      NYY=NY-1
      NY2=NY-2
      PI=3.1415927
      NITER=0
C
C     STEP SIZE IN X AND Y DIRECTIONS. C
```

```
      DX=1./NXX
      DY=1./NYY
      DXX=DX*DX
      DYY=DY*DY
      DO 4 I=1,NX
    4 XX(I)=DX*(I-1)
      DO 6 J=1,NY
6     YY(J)=DY*(J-1)
C
C     IMPOSE STARTING CONDITIONS.
C
      DO 10 I=1,NX
      DO 10 J=1,NY
      PHIO(I,J)=0.0
   10 PHI(I,J)=0.0
C
C     BOUNDARY CONDITIONS
C
      DO 20 I=1,NX
      X=DX*(I-1)
      PHI(I,1)=EXP(0.5*PI*X)
   20 PHI(I,NY)=0.0
      DO 30 J=1,NY
      Y=DY*(J-1)
      PHI(1,J)=COS(0.5*PI*Y)
   30 PHI(NX,J)=EXP(PI*0.5)*COS(0.5*PI*Y)
C
      DO 40 I=2,NXX
      DO 40 J=2,NYY
   40 PHI(I,J)=PHIO(I,J)
C
C     CALCULATE THE LEFT HAND SIDE TERMS.
C
      AX=1./DXX
      BX=-2./DXX
      CX=1./DXX
C
      AY=1./DYY
      BY=-2./DYY
      CY=1./DYY
C
C     CALCULATE THE EXACT SOLUTION.
C
      DO 50 I=1,NX
      DO 50 J=1,NY
   50 PHIEX(I,J)=EXP(0.5*PI*XX(I))*COS(0.5*PI*YY(J))
```

```
C
C     START THE SWEEPS.
C
   60 NITER=NITER+1
      TOTAL=0.0
C
C     X SWEEP.
C
      DO 90 J=2,NYY
C
      DO 70 I=2,NXX
      II=I-1
      R(II)=(PHI(I-1,J)-2.*PHI(I,J)+PHI(I+1,J))/DXX
      R(II)=R(II)+(PHI(I,J-1)-2.*PHI(I,J)+PHI(I,J+1))/DYY
C
      TOTAL=TOTAL+R(II)*R(II)
      R(II)=ALPHA*WOM*R(II)
C
      A(1,II)=0.0
      A(2,II)=-ALPHA*AX
      A(3,II)=1.-ALPHA*BX
      A(4,II)=-ALPHA*CX
      A(5,II)=0.0
   70 CONTINUE
      A(2,1)=0.0
      A(4,NX2)=0.0
      CALL BANFAC(A,NX2,1)
      CALL BANSOL(R,DUM,A,NX2,1)
      DO 80 I=2,NXX
   80 CORR(I,J)=DUM(I-1)
   90 CONTINUE
C
C     RMS OF RHS.
C
      RMS=SQRT(TOTAL/(NXX*NYY))
C
C     Y SWEEP.
C
      DO 120 I=2,NXX
C
      DO 100 J=2,NYY
      JJ=J-1
      R(JJ)=CORR(I,J)
      A(1,JJ)=0.0
      A(2,JJ)=-ALPHA*AY
      A(3,JJ)=1.-ALPHA*BY
      A(4,JJ)=-ALPHA*CY
      A(5,JJ)=0.0
  100 CONTINUE
      A(2,1)=0.0
      A(4,NY2)=0.0
      CALL BANFAC(A,NY2,1)
      CALL BANSOL(R,DUM,A,NY2,1)
      DO 110 J=2,NYY
  110 PHI(I,J)=PHIO(I,J)+DUM(J-1)
  120 CONTINUE
      IF(RMS.LT.RLIMIT) GO TO 200
      IF(NITER.GT.NLIMIT) GO TO 210
      CHG=0.0
      CHE=0.0
      DO 140 I=1,NX
      DO 140 J=1,NY
      CHEE=PHI(I,J)-PHIEX(I,J)
      CHE=CHE+CHEE*CHEE
      CHGG=PHI(I,J)-PHIO(I,J)
      CHG=CHG+CHGG*CHGG
  140 PHIO(I,J)=PHI(I,J)
      CHRMS=SQRT(CHG/(NXX*NYY))
      SOLNER=SQRT(CHE/(NXX*NYY))
      GO TO 60
  200 WRITE(8,*)'CONVERGED RESULTS'
      GO TO 250
C
  210 WRITE(8,*)'CONVERGENCE NOT REACHED'
C
  250 WRITE(8,*)'NO OF ITERATIONS=',NITER
      WRITE(8,*)'RMS CORRECTION=',RMS
      WRITE(8,*)'RMS RHS=',CHRMS
      STOP
      END
```

ii) The approximate factorisation scheme discussed above can be incorporated into a multigrid cycle in the manner indicated in Fig.6.21 of the text and may be summarised as follows:

1. Starting with the fine grid (say 41 x 41) three more grids, (21 x 21), (11 x 11) and (6 x 6) are chosen such that the grid size is successively halved.

2. A relaxation is performed using the procedure described in section (i) with the relation $A^M\phi^M = 0$ on the finest grid.

3. On the next coarser grid , 'm', a relaxation is carried out using,

$$A^m\phi^{m,a} = A^m\phi^m - I^m_{m+1}R^{m+1}, \tag{1}$$

where $\phi^{m,a}$ is the approximate solution sought, ϕ^m is the available solution on the current grid and I^m_{m+1} is the restriction operator for interpolating the fine grid solution to the coarse grid (m).

4. This process is continued until the coarsest grid is reached, i.e. 6 x 6 in the present case.

5. On the coarsest grid, the relaxation procedure is applied to

$$A^1\phi^1 = 0.$$

6. As indicated in Fig. 6.2, it is now necessary to move to successively finer grids. On the next finer grid the relaxation scheme is given by

$$\phi^{m+1,a} = \phi^{m+1} + I^{m+1}_m(\phi^{m,a} - \phi^m) .$$

7. This process of prolongation is continued until the finest grid (41 x 41) is reached.

8. Steps 2 to 7 are repeated until a convergence criterion is satisfied.

A multigrid cycle will be very efficient in that it can give a converged solution in a very small number of iterations. The operation count for a single iteration does increase as the solution is to be processed over several grids. Consider a multigrid cycle consisting of four grids. If the CPU time required on the fine grid is 1 unit, then the total time per multigrid cycle will be

$$N_i\left(W + \frac{W}{4} + \frac{W}{8} + \frac{W}{16}\right)\text{units.}$$

which is equal to 1.4375 units. But the small number of iterations required reduces the CPU time requirement.

CTFD Solutions Manual: Chapter 15

Boundary Layer Flow

15.1 A tridiagonal system of equations to implement (15.98) can be derived as follows. Substituting for $u_j^\lambda, v_j^\lambda, L_y u_j$ and $L_{yy} u_j$ in (15.98), we have

$$\begin{aligned}&\left(\lambda u_j^{n+1}+(1-\lambda)u_j^n\right)\left(\frac{u_j^{n+1}-u_j^n}{\Delta x}\right)+\\&\left(\lambda v_j^{n+1}+(1-\lambda)v_j^n\right)\left(\lambda L_y u_j^{n+1}+(1-\lambda)L_y u_j^n\right)\\&=\lambda\left(u_e u_{ex}\right)^{n+1}+(1-\lambda)\left(u_e u_{ex}\right)^n+\nu\left(\lambda L_{yy}u_j^{n+1}+(1-\lambda)L_{yy}u_j^n\right).\end{aligned}\qquad(1)$$

Starting from the solution at 'n', several iterations are performed to update it to the level 'n+1'. Denoting the intermediate iteration level by k, and writing u_j^λ and v_j^λ as $\lambda u_j^k+(1-\lambda)u_j^n$ and $\lambda v_j^k+(1-\lambda)v_j^n$, we have after simplification

$$\begin{aligned}&u_j^\lambda\left(u_j^{k+1}-u_j^n\right)+v_j^\lambda\left\{\lambda\left(u_{j+1}-u_{j-1}\right)^{k+1}\right.\\&\left.+(1-\lambda)\left(u_{j+1}-u_{j-1}\right)^n\right\}\frac{\Delta x}{2\Delta y}\\&=\Delta x\left\{\lambda\left[u_e u_{ex}\right]^{n+1}+(1-\lambda)\left[u_e u_{ex}\right]^n\right\}\\&+\frac{\nu\Delta x}{\Delta y^2}\left\{\lambda\left(u_{j+1}-2u_j+u_{j-1}\right)^{k+1}+(1-\lambda)\left(u_{j+1}-2u_j+u_{j-1}\right)^n\right\}.\end{aligned}\qquad(2)$$

Substituting $\gamma=0.5v_j^\lambda\Delta x/\Delta y$ and $\delta=\nu\Delta x/\Delta y^2$, we have,

$$\begin{aligned}&u_j^\lambda u_j^{k+1}+\gamma\lambda\left(u_{j+1}-u_{j-1}\right)^{k+1}-\delta\lambda\left(u_{j+1}-2u_j+u_{j-1}\right)^{k+1}\\&=u_j^\lambda u_j^n-\gamma(1-\lambda)\left(u_{j+1}-u_{j-1}\right)^n+\Delta x\lambda\left[u_x u_{ex}\right]^{n+1}\\&+\Delta x(1-\lambda)\left[u_x u_{ex}\right]^n+\delta(1-\lambda)\left(u_{j+1}-2u_j+u_{j-1}\right)^n\end{aligned}\qquad(3)$$

i.e. $\left(-\gamma\lambda-\lambda\delta\right)u_{j-1}^{k+1}+\left(u_j^\lambda+2\delta\lambda\right)u_j^{k+1}+\left(\gamma\lambda-\delta\lambda\right)u_{j+1}^{k+1}$
$=\ d_j\ =RHS\ of\ (3)$,

as required.

15.2 The changes required in program LAMBL to apply the scheme developed in Problem 15.1 are indicated in the listing given below. Two additional arrays UK and VK have been used. The listing shown replaces lines from 59 to 102 of LAMBL.

On running the program for the specified number of points in X and Y directions, the following rms errors in the velocity distribution are obtained.

Case	$JMAX$	Δx	$NMAX$	$\lambda=0.5$	$\lambda=1.0$
(a)	41	0.1	20	0.000455	0.00454
		0.2	10	0.000501	0.00856
		0.4	5	0.000612	0.01480

Case	Δx	$NMAX$	$JMAX$	$\lambda=0.5$	$\lambda=1.0$
(b)	0.02	100	21	0.00064	0.00056
			11	0.00203	0.00144
			6	0.00892	0.00823

In case a) we have a fine mesh in the y direction and the major contribution to the solution error is from the discretisation in the x direction. We find that the errors are almost an order of magnitude smaller for the Crank-Nicolson scheme. Further, the fully implicit scheme, i.e. when λ=1.0, exhibits a first-order convergence, the error decreasing as Δx. For the Crank-Nicolson scheme the error has a poor dependence on Δx. In case b) the mesh is fine in the x direction so that the errors are dominated by those due the discretisation in y direction. Now both the Crank-Nicolson and the fully implicit schemes exhibit a second-order convergence, i.e. the error decreases as Δy^2. It is of interest to note that program LAMBL exhibits a similar behaviour and the numerical values of the error are closer to those observed for the Crank-Nicolson scheme.

Modifications to LAMBL for Problem 15.2

```
      DO 66 J=1,JMAX
      UK(J)=U(J)
   66 VK(J)=V(J)
      UXN=UE
      UEXN=UEX
      DO 10 N = 1,NMAX
      X = X + DX
      UE = X**BETP
      UEX = BETP*UE/X
  555 CONTINUE
      DO 7 J = 2,JMAP
      DY = Y(J) - Y(J-1)
      DYY=DY*DY
      JM = J - 1
C
      ULAM=FLAM*UK(J)+FLAMM*U(J)
      VLAM=FLAM*VK(J)+FLAMM*V(J)
      GAM=0.5*VLAM*DX/DY
      DEL=DX/DYY
      B(2,JM) = -FLAM*(GAM + DEL)
      B(3,JM) = ULAM + 2.*FLAM *DEL
      B(4,JM) = FLAM*(GAM - DEL)
      RHS(JM) = ULAM*U(J)
     1 -0.5*FLAMM*VLAM*(DX/DY)*(U(J+1)-U(J-1))
     2 +DX*FLAM*UE*UEX + DX*FLAMM*UXN*UEXN
     3 +FLAMM*DX*(U(J-1)-2.*U(J)+U(J+1))/DYY
    7 CONTINUE
      RHS(JM) = RHS(JM) - B(4,JM)*UE
      B(4,JM) = 0.
      B(2,1) = 0.
C
C     SOLVE BANDED SYSTEM OF EQUATIONS
C
      CALL BANFAC(B,JM,1)
C
      CALL BANSOL(RHS,UP,B,JM,1)
C
      UP(JMAP) = UE
C
C     OBTAIN V BY INTEGRATING CONTINUITY
C
      DUM = 0.
      SUM = 0.5*(Y(2) - Y(1))
      DO 8 J = 2,JMAX
      DUMH = DUM
      VM(J) = VK(J)
      DY = Y(J) - Y(J-1)
      DUM = 1.5*UP(J-1) - 2.*U(J) + 0.5*UM(J)
      VK(J) = VK(J-1) - 0.5*(DY/DX)*(DUM + DUMH)
    8 CONTINUE
      SUMU=0
      DO 88 J=2,JMAX
```

```
   88 SUMU=SUMU + (UP(J-1)-UK(J))**2
      RMSU = SQRT(SUMU/JMAX)
      IF(RMSU .LT. TOLU) GO TO 89
      DO 87 J=2,JMAX
   87 UK(J)=UP(J-1)
      GO TO 555
   89 CONTINUE
      DO 889 J=2,JMAX
      UM(J) = U(J)
      U(J) = UP(J-1)
      V(J) = VK(J)
      IF(J .EQ. JMAX)GOTO 889
      SUM = SUM + 0.5*(1. - U(J)/UE)*(Y(J+1)-Y(J-1))
  889 CONTINUE
      DISP = SUM/SQRE
      UYZ = (RYP*U(2) - U(3)/RYP)/RY/(Y(2)-Y(1))
      CF = 2.*UYZ/SQRE/UE/UE
      FDD = 0.9278
      DUM = 0.25*X*UE*RE*(2.-BETA)
      EXCF = FDD/SQRT(DUM)
      WRITE(6,9)N,X,EXCF,CF,DISP,UE
    9 FORMAT(' N=',I3,' X=',F4.2,' EXCF=',F9.6,
     1 'CF=',F9.6,2X,' DISP=',F9.6,'  UE=',F6.3)
C
      UXN=UE
      UEXN=UEX
   10 CONTINUE
```

15.3 The following changes are made in LAMBL to compute boundary layer flow over a flat plate.

1. The data statements are changed as suggested.
2. Statements 106 and 107 are replaced so as to calculate EXCF as $0.664(R_{ex})^{-1/2}$.
3. In the data-file, β is set equal to 0.

On running the program, a rms solution error of 0.00483 is obtained. This is larger than for the case, β=0.5, shown in Fig. 15.5. It is also apparent that the skin friction coefficient is not predicted so accurately in the present case, as for β=0.5.

Output for Problem 15.3

```
PROBLEM 15.3 SOLUTION,  BETA= 0.00
 JMAX=  21  DYM=  0.40   RY=  1.00
 NMAX=  19  DX=  0.100E+00  XST=  1.00  RE= 0.100E+06
 N=  1 X=1.20 EXCF= 0.001917  CF= 0.001918  DISP= 0.005974  UE= 1.000
 N=  2 X=1.30 EXCF= 0.001842  CF= 0.001847  DISP= 0.006213  UE= 1.000
 N=  3 X=1.40 EXCF= 0.001775  CF= 0.001779  DISP= 0.006440  UE= 1.000
 N=  4 X=1.50 EXCF= 0.001714  CF= 0.001719  DISP= 0.006661  UE= 1.000
 N=  5 X=1.60 EXCF= 0.001660  CF= 0.001665  DISP= 0.006876  UE= 1.000
 N=  6 X=1.70 EXCF= 0.001610  CF= 0.001616  DISP= 0.007086  UE= 1.000
 N=  7 X=1.80 EXCF= 0.001565  CF= 0.001571  DISP= 0.007289  UE= 1.000
 N=  8 X=1.90 EXCF= 0.001523  CF= 0.001529  DISP= 0.007485  UE= 1.000
 N=  9 X=2.00 EXCF= 0.001485  CF= 0.001490  DISP= 0.007676  UE= 1.000
 N= 10 X=2.10 EXCF= 0.001449  CF= 0.001454  DISP= 0.007861  UE= 1.000
 N= 11 X=2.20 EXCF= 0.001416  CF= 0.001421  DISP= 0.008042  UE= 1.000
 N= 12 X=2.30 EXCF= 0.001385  CF= 0.001390  DISP= 0.008218  UE= 1.000
 N= 13 X=2.40 EXCF= 0.001355  CF= 0.001361  DISP= 0.008389  UE= 1.000
 N= 14 X=2.50 EXCF= 0.001328  CF= 0.001333  DISP= 0.008555  UE= 1.000
 N= 15 X=2.60 EXCF= 0.001302  CF= 0.001307  DISP= 0.008717  UE= 1.000
 N= 16 X=2.70 EXCF= 0.001278  CF= 0.001283  DISP= 0.008875  UE= 1.000
 N= 17 X=2.80 EXCF= 0.001255  CF= 0.001259  DISP= 0.009028  UE= 1.000
 N= 18 X=2.90 EXCF= 0.001233  CF= 0.001237  DISP= 0.009177  UE= 1.000
 N= 19 X=3.00 EXCF= 0.001212  CF= 0.001217  DISP= 0.009322  UE= 1.000
 RMS=  0.483E-02
```

15.4 The change in c_f is chosen as the criterion to adaptively change the value of Δx in LAMBL. The programming logic closely follows lines 122 to 134 of program DOROD. Implementation of this adaptive meshing requires two major changes to be carried out in program LAMBL:

1. To determine the fractional change in c_f at any station and then increase or decrease the step size Δx. If the fractional change in c_f (called $CFCG$) is more than 1.25 times a pre-selected number, $FLOW$ (typically 0.1 or 0.2), then Δx is halved. On the other hand if $CFCG$ is less than 0.75 $\times$ $FLOW$ the mesh size Δx is doubled. The changes in Δx are carried out under the constraint: $\Delta x_{min} \leq \Delta x \leq \Delta x_{max}$.

2. A consequence of changing Δx is that it is no longer uniform. The calculation of LHS and RHS in LAMBL assumes a uniform Δx. Changes are made to generalise these calculations to the case of nonuniform Δx.

The following listing of the program shows the changes made to program LAMBL and replaces lines 60 to 112. An additional array VP is now used to store the temporary updated values of the v velocity. Setting $NCONTR$ =1 causes the program to control the step size Δx. Typical c_f distributions are compared with that for the uniform Δx case in Fig. 1. For the cases where Δx is dynamically changed, values of $\Delta x_{min}, \Delta x_{max}$ and $FLOW$ were prescribed. It is observed that the c_f distribution is close to the exact distribution for all the cases. The computed rms errors for the velocity profiles for the various cases are tabulated below.

Case	rms error
Δx = 0.1, 19 stations	0.000674
Δx = 0.2, 9 stations	0.000765
Δx_{min} =0.01, Δx_{max}=0.2, $FLOW$ = 0.1, 14 stations	0.000487
Δx_{min} =0.01, Δx_{max}=0.3, $FLOW$ = 0.2, 9 stations	0.000354

The table shows that the accuracy is not impaired by changing the step size Δx dynamically. Further, adaptive gridding gives the required accuracy in fewer steps than when a uniform grid is used. This is also demonstrated in the c_f distribution shown in Fig.1.

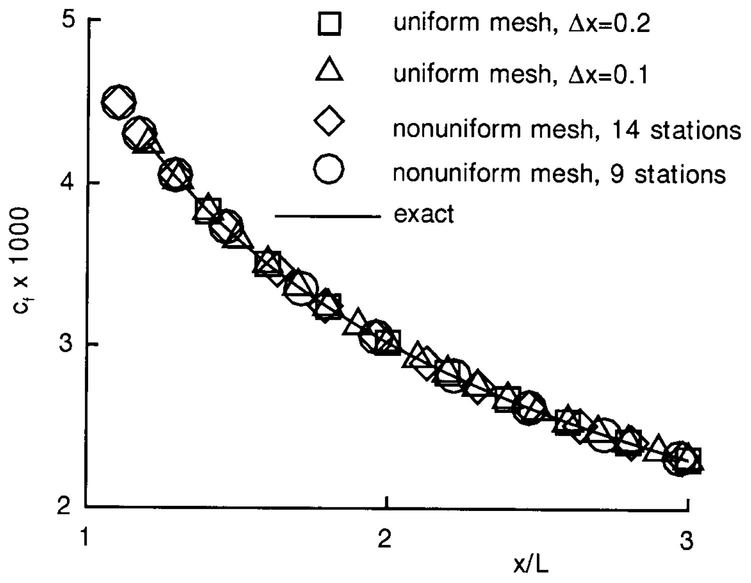

Fig. 1. c_f distribution with uniform and non-uniform meshes in flow direction.

Modifications to LAMBL for Problem 15.4

```
C      PR15P4
C      DX IS VARIED IN RESPONSE TO THE CHANGES IN CF.
C
C      CHANGES IN THE CALCULATION OF LHS AND RHS.
C
```

```
  555 CONTINUE
      X = X + DX
      UE = X**BETP
      UEX = BETP*UE/X
      DO 7 J = 2,JMAP
C
      UC=U(J) + (DX/DX0) *(U(J)-UM(J))
      VC=V(J) + (DX/DX0) *(V(J)-VM(J))
C
      DY = Y(J) - Y(J-1)
      JM = J - 1
      P = VC*DX/RYP/DY
      Q = 2.*DX/(RYP*DY*DY)
      B(2,JM) = -P*RY - Q
      B(3,JM) = 1.5*UC + Q*RYP/RY + P*(RY-1./RY)
      B(4,JM) =   P/RY - Q/RY
      RHS(JM) = UE*UEX*DX + 0.5*(DX/DX0)*UC*(U(J)-UM(J))
     1 + 1.5*UC*U(J)
    7 CONTINUE
      RHS(JM) = RHS(JM) - B(4,JM)*UE
      B(4,JM) = 0.
      B(2,1) = 0.
C
C     SOLVE BANDED SYSTEM OF EQUATIONS
C
      CALL BANFAC(B,JM,1)
C
      CALL BANSOL(RHS,UP,B,JM,1)
C
      UP(JMAP) = UE
C
C     OBTAIN V BY INTEGRATING CONTINUITY
C
      DUM = 0.
      VP(1)=0.0
      SUM = 0.5*(Y(2) - Y(1))
      DO 8 J = 2,JMAX
      DUMH = DUM
      DY = Y(J) - Y(J-1)
      DUM = 1.5*(UP(J-1) - U(J)) -0.5*(DX/DX0)
     1 *(U(J)-UM(J))
      VP(J) = VP(J-1) - 0.5*(DY/DX)*(DUM + DUMH)
      IF(J .EQ. JMAX)GOTO 8
      SUM = SUM + 0.5*(1. - U(J)/UE)*(Y(J+1)-Y(J-1))
    8 CONTINUE
      DISP = SUM/SQRE
```

```
C
C     DECIDE WHETHER DX IS TO BE CHANGED.
C
      IF(NCONTR .EQ. 0) GO TO 110
      UYZ = (RYP*UP(1) - UP(2)/RYP)/RY/(Y(2)-Y(1))
      CF = 2.*UYZ/SQRE/UE/UE
C
      CFCG=ABS(CF-CF0)/CF0
      IF(0.5*DX .LT. DXMIN) GO TO 121
      IF(CFCG .GT. 1.25*FLOW) THEN
      X=X-DX
      DX =0.5*DX
      GO TO 555
      ENDIF
  121 CONTINUE
      DX0=DX
      IF(1.5*DX .GT. DXMAX) GO TO 122
      IF(CFCG .LT. 0.75*FLOW)  DX=1.5*DX
  122 CONTINUE
  110 CONTINUE
      DO 123 J=2,JMAX
      UM(J)=U(J)
      VM(J)=V(J)
      V(J)=VP(J)
  123 U(J)=UP(J-1)
   10 CONTINUE
```

15.5 The modification to LAMBL to adaptively extend y_{max} is carried out as follows. A study of the output from LAMBL shows that y_{max} may be varied according to the formula:

$$y_{max} = y_{max,\,0} + 1.6\big(1000\delta^* - 3.3\big) . \tag{1}$$

and where a nonuniform mesh is used in the y-direction, the mesh spacing close to the wall may be varied as follows:

$$\varDelta y_{min} = \varDelta y_{min,0} + 0.8\big(1000\delta^* - 3.3\big) . \tag{2}$$

The numerical constants appearing in the above equations were obtained after establishing the rate of growth of boundary layer for a few cases and are problem dependent.

In the modified code, the new value of y_{max} is evaluated after the solution is calculated at a station. The distribution of grid points in the y direction is then calculated keeping $JMAX$ the same. A uniform distribution presents no problem. But when the grid points are to be distributed nonuniformly, a new subroutine MERAT is called. This subroutine calculates the value of r_y given

Δy_{min} and y_{max} by iteration. Once r_y is known, the grid point distribution is easily obtained.

As the next step, values of $u^{n-1}, v^{n-1}, u^n, v^n$ (UM, VM, U and V in the code) are interpolated in subroutine INTERP onto the new y-grid. A listing of the modifications to program LAMBL is given below along with those for the new subroutines MERAT and INTERP. The changes to LAMBL are an insert after line 111. When variable $NADAPT$ =1 the program changes y_{max} adaptively. To facilitate interpolation of velocities on to the new y-grid, new arrays $YNEW$, $UNEW$ and $VNEW$ have been used. The outputs from the modified program are also enclosed. In the first case, the mesh in the y direction is uniform and in the second case it is nonuniform. It is observed that y_{max} ranges from 6 to 8.043, and from 7 to 9.378 respectively. The rms error values are now higher than for the case where y_{max} is uniform. The error also increases for the case where the mesh in the y-direction is nonuniform. This is due to the interpolation of the velocity values onto the new y-grid at every station.

Modifications to LAMBL for Problem 15.5

```
C     PR15P5
C
      IF(NADAPT .EQ. 0) GO TO 10
C
C     CALCULATE NEW Y VALUES
C
      YMAX = YMAX0+ 1.6*(1000.0*DISP - 3.3)
      IF(RY .GT. 1.0) THEN
      DYMN = DYM+ .8*(1000.0*DISP - 3.3)
      CALL MERAT(RYY,YMAX,JMAX,DYMN)
      YNEW(1) = 0.
      DY = DYMN/RYY
      DO 302 J = 2,JMAX
      DY = DY*RYY
      YNEW(J) = YNEW(J-1) + DY
  302 CONTINUE
      ELSE
      DDY = YMAX /JMAP
      YNEW(1)=0.0
      DO 301 J=2,JMAX
  301 YNEW(J)=YNEW(J-1) + DDY
C
      ENDIF
C
C     INTERPOLATE THE UAND V VALUES TO THE NEW GRID
C
      CALL INTERP(JMAX,Y,U,YNEW,UNEW)
      CALL INTERP(JMAX,Y,V,YNEW,VNEW)
```

```
      CALL INTERP(JMAX,Y,UM,YNEW,UMEW)
      CALL INTERP(JMAX,Y,VM,YNEW,VMEW)
C
      DO 303 J=1,JMAX
      Y(J)=YNEW(J)
      U(J)=UNEW(J)
      UM(J)=UMEW(J)
      V(J)=VNEW(J)
  303 VM(J)=VMEW(J)
   10 CONTINUE

      Subroutine INTERP(JMAX,YG,UG,Y,U)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION UG(41),YG(41),Y(41),U(41)
      JMAP=JMAX-1
      U(1)=0.0
      U(JMAX)=UG(JMAX)
      DO 10 J=2,JMAP
      DO 5 K=2,JMAP
      IF(Y(J) .GT. YG(K) .AND. Y(J) .LE. YG(K+1)) GO TO 4
      GO TO 5
    4 GRAD =(UG(K+1) - UG(K))/(YG(K+1)-YG(K))
      U(J)=UG(K) + GRAD*(Y(J) - YG(K))
    5 CONTINUE
   10 CONTINUE
      RETURN
      END

      Subroutine MERAT(R,YMAX,JMAX,DYMN)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      JMAP=JMAX-1
      AMAPP=1./JMAP
      R=1.001
    5 ARGG=(DYMN -YMAX  + YMAX*R)/DYMN
      RH=EXP(AMAPP*LOG(ARGG))
      IF(ABS(RH-R).LT.10E-06) GO TO 999
      R=RH
      GO TO 5
  999 RETURN
      END
```

Sample output for Problem 15.5
Uniform mesh

```
FALKNER-SKAN SOLUTION,  BETA= 0.50
JMAX=  21  DYM=  0.30   RY=  1.00
NMAX=  19  DX=  0.100E+00  XST=  1.00  RE= 0.100E+06
```

```
N= 1 X=1.20 EXCF=0.004243 CF=0.004243 DISP=0.003331 UE=1.063 YMAX=6.000
N= 2 X=1.30 EXCF=0.004022 CF=0.004021 DISP=0.003423 UE=1.091 YMAX=6.051
N= 3 X=1.40 EXCF=0.003828 CF=0.003824 DISP=0.003515 UE=1.119 YMAX=6.204
N= 4 X=1.50 EXCF=0.003656 CF=0.003649 DISP=0.003603 UE=1.145 YMAX=6.359
N= 5 X=1.60 EXCF=0.003502 CF=0.003493 DISP=0.003686 UE=1.170 YMAX=6.505
N= 6 X=1.70 EXCF=0.003364 CF=0.003353 DISP=0.003766 UE=1.193 YMAX=6.644
N= 7 X=1.80 EXCF=0.003238 CF=0.003226 DISP=0.003842 UE=1.216 YMAX=6.776
N= 8 X=1.90 EXCF=0.003123 CF=0.003111 DISP=0.003915 UE=1.239 YMAX=6.903
N= 9 X=2.00 EXCF=0.003018 CF=0.003005 DISP=0.003985 UE=1.260 YMAX=7.025
N=10 X=2.10 EXCF=0.002922 CF=0.002908 DISP=0.004053 UE=1.281 YMAX=7.142
N=11 X=2.20 EXCF=0.002832 CF=0.002819 DISP=0.004119 UE=1.301 YMAX=7.255
N=12 X=2.30 EXCF=0.002750 CF=0.002736 DISP=0.004182 UE=1.320 YMAX=7.364
N=13 X=2.40 EXCF=0.002673 CF=0.002659 DISP=0.004243 UE=1.339 YMAX=7.470
N=14 X=2.50 EXCF=0.002601 CF=0.002587 DISP=0.004303 UE=1.357 YMAX=7.572
N=15 X=2.60 EXCF=0.002534 CF=0.002520 DISP=0.004361 UE=1.375 YMAX=7.672
N=16 X=2.70 EXCF=0.002471 CF=0.002457 DISP=0.004417 UE=1.392 YMAX=7.768
N=17 X=2.80 EXCF=0.002412 CF=0.002398 DISP=0.004472 UE=1.409 YMAX=7.862
N=18 X=2.90 EXCF=0.002356 CF=0.002342 DISP=0.004526 UE=1.426 YMAX=7.954
N=19 X=3.00 EXCF=0.002303 CF=0.002290 DISP=0.004578 UE=1.442 YMAX=8.043
RMS= 0.478E-02
```

Non uniform mesh

```
FALKNER-SKAN SOLUTION, BETA= 0.50
JMAX= 21 DYM= 0.30 RY= 1.02
NMAX= 19 DX= 0.100E+00 XST= 1.00 RE= 0.100E+06
N= 1 X=1.20 EXCF=0.004243 CF=0.004243 DISP=0.003333 UE=1.063 YMAX=7.289
N= 2 X=1.30 EXCF=0.004022 CF=0.004021 DISP=0.003425 UE=1.091 YMAX=7.340
N= 3 X=1.40 EXCF=0.003828 CF=0.003822 DISP=0.003519 UE=1.119 YMAX=7.493
N= 4 X=1.50 EXCF=0.003656 CF=0.003645 DISP=0.003608 UE=1.145 YMAX=7.649
N= 5 X=1.60 EXCF=0.003502 CF=0.003486 DISP=0.003694 UE=1.170 YMAX=7.799
N= 6 X=1.70 EXCF=0.003364 CF=0.003344 DISP=0.003776 UE=1.193 YMAX=7.941
N= 7 X=1.80 EXCF=0.003238 CF=0.003216 DISP=0.003854 UE=1.216 YMAX=8.078
N= 8 X=1.90 EXCF=0.003123 CF=0.003099 DISP=0.003929 UE=1.239 YMAX=8.209
N= 9 X=2.00 EXCF=0.003018 CF=0.002993 DISP=0.004002 UE=1.260 YMAX=8.333
N=10 X=2.10 EXCF=0.002922 CF=0.002895 DISP=0.004071 UE=1.281 YMAX=8.453
N=11 X=2.20 EXCF=0.002832 CF=0.002805 DISP=0.004138 UE=1.301 YMAX=8.569
N=12 X=2.30 EXCF=0.002750 CF=0.002721 DISP=0.004204 UE=1.320 YMAX=8.680
N=13 X=2.40 EXCF=0.002673 CF=0.002643 DISP=0.004267 UE=1.339 YMAX=8.789
N=14 X=2.50 EXCF=0.002601 CF=0.002571 DISP=0.004328 UE=1.357 YMAX=8.894
N=15 X=2.60 EXCF=0.002534 CF=0.002503 DISP=0.004388 UE=1.375 YMAX=8.997
N=16 X=2.70 EXCF=0.002471 CF=0.002440 DISP=0.004446 UE=1.392 YMAX=9.096
N=17 X=2.80 EXCF=0.002412 CF=0.002381 DISP=0.004502 UE=1.409 YMAX=9.192
N=18 X=2.90 EXCF=0.002356 CF=0.002325 DISP=0.004557 UE=1.426 YMAX=9.286
N=19 X=3.00 EXCF=0.002303 CF=0.002272 DISP=0.004611 UE=1.442 YMAX=9.378
RMS= 0.671E-02
```

15.6 a) The truncation error analysis for (15.17) is carried out by expanding every term on the RHS as a Taylor series about y_j. The first equation in (15.17) yields,

$$\frac{\partial u}{\partial y} = u_y + r_y\frac{\Delta y^2}{6}u_{y^3} + r_y(r_y-1)\frac{\Delta y^3}{24}u_{y^4} + O(H)\ . \tag{1}$$

Thus, the approximation for $\partial u/\partial y$ is second-order accurate.

b) The same analysis for the second equation of (15.17) gives,

$$\begin{aligned}\frac{\partial^2 u}{\partial y^2} =& \frac{2}{(1+r_y)\Delta y^2}\Big\{ u - u_y\Delta y + \frac{\Delta y^2}{2}u_{yy} - \frac{\Delta y^3}{6}u_{y^3} \\ &+ \frac{\Delta y^4}{24}u_{y^4} + \frac{u}{r_y} + \Delta y\ u_y + r_y\frac{\Delta y^2}{2}u_{yy} + r_y^2\frac{\Delta y^3}{6}u_{y^3} \\ &+ r_y^3\frac{\Delta y^4}{24}u_{y^4} - (1+\frac{1}{r_y})u\Big\} \\ =& \ u_{yy} + \frac{(r_y-1)}{3}\Delta y\ u_{y^3} + \frac{(r_y^3+1)\Delta y^2}{12}u_{y^4} + O(H)\ .\end{aligned} \tag{2}$$

Thus the expression for $\partial^2 u/\partial y^2$ is only first-order accurate, but it becomes second-order accurate when $r_y = 1+\text{O}(\Delta\text{y})$ i.e. when the spacing is close to uniform.

c) Now, consider the approximation,

$$\frac{\partial u}{\partial y} \approx \frac{u_{j+1}-u_{j-1}}{(1+r_y)\Delta y}\ . \tag{3}$$

A truncation error analysis of (3) gives,

$$\frac{\partial u}{\partial y} = u_y + (r_y-1)\frac{\Delta y}{2}u_{yy} + \frac{(r_y^3+1)}{r_y+1}\frac{\Delta y^2}{6}u_{y^3} + O(H)\ . \tag{4}$$

Thus the approximation is only first-order accurate. Using (3), for a near uniform grid (i.e. $r_y = 1+\text{O}(\Delta y)$, second-order accuracy is obtained.

d) Expanding the terms of (3) as a Taylor series about a point mid way between $(j-1)$ and $(j+1)$ i.e. at $y + \frac{\Delta y}{2}(r_y-1)$ gives,

$$\begin{aligned}\frac{\partial u}{\partial y} \approx& \frac{1}{(1+r_y)\Delta y}\Big\{ \\ & u + (\frac{1+r_y}{2})\Delta y u_y + (\frac{1+r_y}{2})^2\frac{\Delta y^2}{2}u_{yy} + (\frac{1+r_y}{2})^3\frac{\Delta y^3}{6}u_{y^3} \\ & - u + (\frac{1+r_y}{2})\Delta y u_y - (\frac{1+r_y}{2})^2\frac{\Delta y^2}{2}u_{yy} + (\frac{1+r_y}{2})^3\frac{\Delta y^3}{6}u_{y^3}\Big\} \\ \approx& \ u_y + \frac{(r_y^3+1)}{r_y+1}\frac{\Delta y^2}{6}u_{y^3} + O(H)\ .\end{aligned} \tag{5}$$

Now, we see that the approximation given by (3) is second-order accurate. It appears that second-order accuracy may be achieved for the convective terms by suitably centering the velocity differences, eg., mid way between (j-1) and (j+1). For (15.5) as a whole it may be possible to find an intermediate point where truncation error contribution from $(u\partial u/\partial y)$ and $\partial^2 u/\partial y^2$ cancel.

15.7 Starting from (15.27) and (15.28) and substituting for the various terms from (15.25) and (15.26), we have after noting that for a flat plate $du_e/dx = 0$,

$$\frac{0.5}{\Delta x}\left(u_{j-1}^{n+1}+u_j^{n+1}-u_j^n-u_{j-1}^n\right)+\frac{1}{\Delta y}\left(-v_{j-1}^{n+1}+v_j^{n+1}+v_j^n-v_{j-1}^n\right) = 0\ , \qquad (1)$$

$$\begin{aligned}&\frac{1}{16\Delta x}\left(u_{j-1}^{n+1}+u_j^{n+1}+u_j^n+u_{j-1}^n\right)\left(u_{j-1}^{n+1}+u_j^{n+1}-u_j^n-u_{j-1}^n\right)\\ &+\frac{1}{16\Delta y}\left(v_{j-1}^{n+1}+v_j^{n+1}+v_j^n+v_{j-1}^n\right)\left(-u_{j-1}^{n+1}+u_j^{n+1}+u_j^n-u_{j-1}^n\right)\\ &=\frac{0.5}{\Delta y}\left(\tau_j^n-\tau_{j-1}^n+\tau_j^{n+1}-\tau_{j-1}^{n+1}\right)\ , \quad \text{and}\end{aligned} \qquad (2)$$

$$0.5(\tau_j^{n+1}+\tau_{j-1}^{n+1}) = 0.5(\nu_j^{n+1}+\nu_{j-1}^{n+1})(u_j^{n+1}-u_{j-1}^{n+1})/\Delta y\ . \qquad (3)$$

To render these equations suitable for Newton's method, we replace the "n+1" terms in the coefficients of the above equations by "k", and simplify, to get

$$\Delta u_j^{k+1}+\Delta u_{j-1}^{k+1}+2\frac{\Delta x}{\Delta y}(\Delta v_j^{k+1}-\Delta v_{j-1}^{k+1}) = -4\frac{\Delta x}{\Delta y}(v_j^n+v_{j-1}^n)\ , \qquad (4)$$

$$\begin{aligned}&\left(\frac{0.25U_c}{\Delta x}+\frac{0.5V_c}{\Delta y}\right)\Delta u_j^{k+1} + \left(\frac{0.25U_c}{\Delta x}-\frac{0.5V_c}{\Delta y}\right)\Delta u_{j-1}^{k+1}\\ &-\frac{2}{\Delta y}\left(\Delta\tau_j^{k+1}-\Delta\tau_{j-1}^{k+1}\right) = \frac{4}{\Delta y}\left(\tau_j^n-\tau_{j-1}^n\right)\ ,\end{aligned} \qquad (5)$$

and

$$\begin{aligned}&-\frac{0.5}{\Delta y}\left(\nu_j^n+\nu_{j-1}^n\right)\left(\Delta u_j^{k+1}-\Delta u_{j-1}^{k+1}\right) + 0.5\left(\Delta\tau_j^{k+1}+\Delta\tau_{j-1}^{k+1}\right)\\ &= -0.5\left(\tau_j^n+\tau_{j-1}^n\right) + \frac{0.5}{\Delta y}\left(\nu_j^n+\nu_{j-1}^n\right)\left(u_j^n-u_{j-1}^n\right)\ .\end{aligned} \qquad (6)$$

where

$$U_c = u_j^k+u_{j-1}^k+u_j^n+u_{j-1}^n \quad \text{and} \quad V_c = v_j^k+v_{j-1}^k+v_j^n+v_{j-1}^n\ .$$

Thus if j=1 denotes the wall and $j = JM$ denotes the outer edge of the computational domain one has to solve the following equations.

$$\Delta u_1^{k+1} = 0\ , \qquad (7)$$

$$\Delta v_1^{k+1} = 0\ , \qquad (8)$$

$$\Delta u_2^{k+1} - F_1\Delta v_2^{k+1} = 2F_1 v_2^n\ . \qquad (9)$$

In (9) the continuity equation has been written as if it were a boundary condition.

For j =2,3,4,..........,$JM-1$, we have,

$$F_3\Delta u_j^{k+1}+F_2\Delta u_{j-1}^{k+1}-\frac{2}{\Delta y}\Delta\tau_j^{k+1}+\frac{2}{\Delta y}\Delta\tau_{j-1}^{k+1} = \frac{4}{\Delta y}(\tau_j^n-\tau_{j-1}^n)\ , \qquad (10)$$

$$-F_4\Delta u_j^{k+1}+F_4\Delta u_{j-1}^{k+1}+0.5\Delta\tau_j^{k+1}+0.5\Delta\tau_{j-1}^{k+1} = RHS\ of\ (6)\ , \qquad (11)$$

$$\Delta u_{j+1}^{k+1}+\Delta u_j^{k+1}-F_1\Delta v_{j+1}^{k+1}+F_1\Delta v_j^{k+1} = -2F_1(v_{j+1}^n+v_j^n)\ . \qquad (12)$$

At $j = JM$, we have

$$\begin{aligned}&F_3\Delta u_{JM}^{k+1}+F_2\Delta u_{JM-1}^{k+1}-\frac{2}{\Delta y}\Delta\tau_{JM}^{k+1}+\frac{2}{\Delta y}\Delta\tau_{JM-1}^{k+1}\\ &=\frac{4}{\Delta y}(\tau_{JM}^n-\tau_{JM-1}^n)\ ,\end{aligned} \qquad (13)$$

$$\begin{aligned}&-F_4\Delta u_{JM}^{k+1}+F_4\Delta u_{JM-1}^{k+1}+0.5\Delta\tau_{JM}^{k+1}+0.5\Delta\tau_{JM-1}^{k+1}\\ &= RHS\ of\ (6)\ \text{for}\ j = JM\ ,\end{aligned} \qquad (14)$$

$$\Delta u_{JM}^{k+1} = 0\ , \qquad (15)$$

where

$$\begin{aligned}F_1 &= -2\Delta x/\Delta y,\\ F_2 &= (.25U_c/\Delta x)-(.5V_c/\Delta y),\\ F_3 &= (.25U_c/\Delta x)+(.5V_c/\Delta y),\\ F_4 &= (0.5/\Delta y)(\nu_j^n+\nu_{j-1}^n).\end{aligned}$$

These equations are written in the matrix form as

$$B_1W_1^{k+1}-C_1W_2^{k+1} = D_1^n\ , \qquad (16)$$

$$A_jW_{j-1}^{k+1}+B_jW_j^{k+1}+C_jW_{j+1}^{k+1} = D_j^n\ , \qquad (17)$$

$$A_{JM}W_{JM-1}^{k+1}+B_{JM}W_{JM}^{k+1} = D_{JM}^n\ , \qquad (18)$$

where ,

$$W = \left\{\Delta u^{k+1},\Delta v^{k+1},\Delta\tau^{k+1}\right\}^T \quad \text{and} \qquad (19)$$

and typically,

$$A_j = \left\{\begin{matrix} F_2 & 0 & 2/\Delta y \\ F_4 & 0 & 0.5 \\ 0 & 0 & 0\end{matrix}\right\}, B_j = \left\{\begin{matrix} F_3 & 0 & -2/\Delta y \\ -F_4 & 0 & 0.5 \\ 1 & F_1 & 0\end{matrix}\right\}\ ,$$

$$C_j = \begin{Bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & -F_1 & 0 \end{Bmatrix} \quad \text{and} \quad D_j = \begin{Bmatrix} 4(\tau_j^n - \tau_{j-1}^n)/\Delta y \\ RHS\ of\ (6) \\ -2F_1(v_j^n + v_{j+1}^n) \end{Bmatrix} . \tag{20}$$

These equations are solved using any block triagonal solver (e.g., the one described in Sect. 6.2.5) for corrections at the iteration level k.

15.8 On replacing the term

$0.5(u_j^n + u_j^{n+1})\ (u_j^{n+1} - u_j^n)/\Delta x$ in (15.101) by

$0.5(u_j^n + u_j^k)\ (u_j^{n+1} - u_j^n)/\Delta x$,

and linearising $v_j^{n+1}(u_{j+1}^{n+1} - u_{j-1}^{n+1})$ as

$v_j^{n+1}(u_{j+1}^n - u_{j-1}^n) + v_j^n(u_{j+1}^{n+1} - u_{j-1}^{n+1}) - v_j^n(u_{j+1}^n - u_{j-1}^n)$,

one gets

$$\begin{aligned} &(u_j^n + u_j^k)(u_j^{k+1} - u_j^n) \\ &+ 0.5\frac{\Delta x}{\Delta y}\left\{v_j^k(u_{j+1}^{k+1} - u_{j-1}^{k+1}) + v_j^{k+1}(u_{j+1}^k - u_{j-1}^k) - v_j^k(u_{j+1}^k - u_{j-1}^k)\right\} \\ &- \ 0.5\frac{\Delta x}{\Delta y^2}\left(u_{j-1}^{k+1} - 2u_j^{k+1} + u_{j+1}^{k+1}\right) \ - \ 0.5\frac{\Delta x}{\Delta y^2}\left(u_{j-1}^n - 2u_j^n + u_{j+1}^n\right) \\ &= 0 \ . \end{aligned} \tag{1}$$

This formula is rearranged to give

$$a_j^k u_{j-1}^{k+1} \ + \ b_j^k u_j^{k+1} \ + \ c_j^k u_{j+1}^{k+1} \ + \ g_j^k v_j^{k+1} \ = \ d_j^k$$

where,

$$\begin{aligned} a_j^k \ &= \ -0.5\frac{\Delta x}{\Delta y}v_j^k - 0.5\frac{\Delta x}{\Delta y^2}, \quad b_j^k \ = \ u_j^n + u_j^k + \frac{\Delta x}{\Delta y^2} \ , \\ c_j^k \ &= \ 0.5\frac{\Delta x}{\Delta y}v_j^k - 0.5\frac{\Delta x}{\Delta y^2} \ , \quad g_j^k \ = \ 0.5\frac{\Delta x}{\Delta y}(u_{j+1}^k - u_{j-1}^k) \ , \\ \text{and} \quad d_j^k \ &= \ 0.5\frac{\Delta x}{\Delta y^2}(u_{j+1}^n + u_{j-1}^n) \ + \ u_j^n(u_j^k + u_j^n - \Delta x/\Delta y^2) \\ &\quad + \ 0.5\frac{\Delta x}{\Delta y}\left[v_j^k(u_{j+1}^k - u_{j-1}^k) - v_j^n(u_{j+1}^n - u_{j-1}^n)\right] . \end{aligned} \tag{2}$$

Writing the continuity equation (15.100) as an algorithm we have

$$v_j^{k+1} \ = \ -\frac{\Delta y}{\Delta x}(u_j^{k+1} - u_j^n + u_{j-1}^{k+1} - u_{j-1}^n) + v_{j-1}^{k+1} + v_{j-1}^n - v_j^n \ . \tag{3}$$

15.9 To execute the Davis coupled scheme (Sect. 15.2.3) for boundary layer flow past a flat plate it is necessary to modify program LAMBL in the following way. First the values of UB , VB and YZ given in Problem 15.3 are substituted

for lines 6 to 13. The coefficients B and the extra coefficients are determined by the following code which replaces lines 66 tp 78. A new subroutine, COUPLED, replaces the call to BANFAC and BANSOL and the subsequent calculation of v, i.e. lines 79 to 100, are replaced with the following code,

Modifications to LAMBL for Problem 15.9

```
C     PR15P9.
C
  555 CONTINUE
       KCT = KCT + 1
       DO 7 J = 2,JMAX
       DY = Y(J) - Y(J-1)
       DXY=DX/DY
       DEL=DX/DY/DY
       IF(J .EQ. JMAX)GOTO 69
C
       B(1,J) = -0.5*DXY*VK(J) -     DEL
       B(3,J) =  0.5*DXY*VK(J) -     DEL
       B(2,J) =   U(J) +  UK(J) + 2.*DEL
       B(4,J) = 0.5*DXY*(UK(J+1)-UK(J-1))
       RHS(J) =   DEL * (U(J+1) + U(J-1) -2.*U(J))
      1 + U(J) *U(J) +U(J)*UK(J) -0.5*DXY*V(J)*(U(J+1)-U(J-1))
      2 + 0.5*DXY*VK(J)*(UK(J+1)-UK(J-1)) + 2.0*DX*UEH*UEX
C
   69  T(J) = -V(J) + V(J-1)  + DYX *(U(J) + U(J-1))
       S(J)= 1.0/DXY
     7 CONTINUE
       RHS(JMAP) = RHS(JMAP) - B(3,JMAP)*UE
       B(3,JMAP) = 0.
       B(1,2) = 0.
C
C      SOLVE COUPLED BANDED SYSTEM OF EQUATIONS
C
       CALL COUPLED(JMAX,B,T,S,RHS,UP,VP)
C
       SUMV=0.
       DO 88 J=2,JMAX
    88 SUMV=SUMV + (VP(J)-VK(J))**2
       RMSV = SQRT(SUMV/JMAX)
       IF(RMSV .LT. TOLV) GO TO 89
       IF(KCT .GT. KCMAX)GOTO 89
       DO 87 J=2,JMAX
       VK(J) = VP(J)
       UK(J) = UP(J)
   87 CONTINUE
      GO TO 555
   89 CONTINUE
      SUM = 0.
      DO 889 J=1,JMAX
      U(J) = UP(J)
      V(J) = VP(J)
      UK(J) = U(J)
      VK(J) = V(J)
      IF(J .EQ. 1)GOTO 889
      SUM = SUM + (1. - 0.5*(U(J-1)+U(J))/UE)*(Y(J)-Y(J-1))
  889 CONTINUE

      Subroutine COUPLED(JMAX,BB,TT,S,D,U,V)
      DIMENSION BB(4,65),TT(65),S(65),U(65),V(65)
     1 ,A(65),B(65),C(65),D(65),G(65),E(65),SE(65),GG(65)
       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      E(JMAX)=0.0
      GG(JMAX)=0.0
      SE(JMAX)=0.0
      JM=JMAX -1
C
      DO 5 J=2,JM
      A(J)=BB(1,J)
      B(J)=BB(2,J)
      C(J)=BB(3,J)
    5 G(J)=BB(4,J)
      DO 10 J = JM ,2,-1
      FACT = C(J)*GG(J+1) + G(J)
      T = B(J) + C(J)*E(J+1) - S(J)*FACT
      E(J)= -(A(J) -S(J)*FACT)/T
      GG(J)=-FACT/T
   10 SE(J)= (D(J) -FACT*TT(J) - C(J)*SE(J+1))/T
C
      DO 20 J=2,JMAX
      IF(J .EQ. JMAX)GOTO 20
      U(J)= E(J)*U(J-1) + GG(J)*V(J-1) + SE(J)
   20 V(J) =V(J-1) - S(J)*(U(J-1)+U(J)) + TT(J)
      RETURN
      END
```

Typical results for the Davis coupled scheme and the original LAMBL scheme are shown in the tables. It is apparent that both the schemes are producing accurate results for c_f and the displacement thickness, δ^*.

Results for Davis coupled scheme

```
FLAT PLATE (DAVIS) SOLUTION,  BETA= 0.00
JMAX,KCMAX=  21 10  DYM=  0.40   RY=  1.00  TOLV= 0.100E-05
NMAX = 20  DX=  0.500E-01  XST=  1.00  RE= 0.100E+06
N=  1 X=1.05 EXCF= 0.002049  CF= 0.002059   EXDIS= 0.005575 DISP= 0.005590
N=  2 X=1.10 EXCF= 0.002001  CF= 0.002001   EXDIS= 0.005706 DISP= 0.005723
N=  3 X=1.15 EXCF= 0.001958  CF= 0.001962   EXDIS= 0.005834 DISP= 0.005849
N=  4 X=1.20 EXCF= 0.001916  CF= 0.001918   EXDIS= 0.005959 DISP= 0.005975
N=  5 X=1.25 EXCF= 0.001878  CF= 0.001881   EXDIS= 0.006082 DISP= 0.006097
N=  6 X=1.30 EXCF= 0.001841  CF= 0.001844   EXDIS= 0.006203 DISP= 0.006216
N=  7 X=1.35 EXCF= 0.001807  CF= 0.001810   EXDIS= 0.006321 DISP= 0.006334
N=  8 X=1.40 EXCF= 0.001774  CF= 0.001777   EXDIS= 0.006437 DISP= 0.006449
N=  9 X=1.45 EXCF= 0.001743  CF= 0.001746   EXDIS= 0.006551 DISP= 0.006562
N= 10 X=1.50 EXCF= 0.001714  CF= 0.001717   EXDIS= 0.006663 DISP= 0.006673
N= 11 X=1.55 EXCF= 0.001686  CF= 0.001689   EXDIS= 0.006773 DISP= 0.006783
N= 12 X=1.60 EXCF= 0.001660  CF= 0.001663   EXDIS= 0.006881 DISP= 0.006890
N= 13 X=1.65 EXCF= 0.001634  CF= 0.001637   EXDIS= 0.006988 DISP= 0.006996
N= 14 X=1.70 EXCF= 0.001610  CF= 0.001613   EXDIS= 0.007093 DISP= 0.007100
N= 15 X=1.75 EXCF= 0.001587  CF= 0.001590   EXDIS= 0.007196 DISP= 0.007203
N= 16 X=1.80 EXCF= 0.001565  CF= 0.001568   EXDIS= 0.007298 DISP= 0.007304
N= 17 X=1.85 EXCF= 0.001543  CF= 0.001546   EXDIS= 0.007399 DISP= 0.007404
N= 18 X=1.90 EXCF= 0.001523  CF= 0.001526   EXDIS= 0.007498 DISP= 0.007501
N= 19 X=1.95 EXCF= 0.001503  CF= 0.001506   EXDIS= 0.007596 DISP= 0.007598
N= 20 X=2.00 EXCF= 0.001485  CF= 0.001487   EXDIS= 0.007693 DISP= 0.007693
RMS=  0.756E-03
```

Results for LAMBL scheme

```
FLAT PLATE (LAMBL) SOLUTION,  BETA= 0.00
JMAX=  21  DYM=  0.40   RY=  1.00
NMAX=  20 DX=  0.500E-01 XST=  0.95 RE= 0.100E+06
N=  1 X=1.05 EXCF= 0.002049  CF= 0.002050   EXDIS= 0.005573 DISP= 0.005591
N=  2 X=1.10 EXCF= 0.002002  CF= 0.002007   EXDIS= 0.005705 DISP= 0.005720
N=  3 X=1.15 EXCF= 0.001958  CF= 0.001963   EXDIS= 0.005833 DISP= 0.005846
N=  4 X=1.20 EXCF= 0.001917  CF= 0.001920   EXDIS= 0.005958 DISP= 0.005970
N=  5 X=1.25 EXCF= 0.001878  CF= 0.001881   EXDIS= 0.006081 DISP= 0.006092
N=  6 X=1.30 EXCF= 0.001842  CF= 0.001845   EXDIS= 0.006202 DISP= 0.006213
N=  7 X=1.35 EXCF= 0.001807  CF= 0.001810   EXDIS= 0.006320 DISP= 0.006331
N=  8 X=1.40 EXCF= 0.001775  CF= 0.001778   EXDIS= 0.006436 DISP= 0.006445
N=  9 X=1.45 EXCF= 0.001744  CF= 0.001747   EXDIS= 0.006550 DISP= 0.006557
N= 10 X=1.50 EXCF= 0.001714  CF= 0.001718   EXDIS= 0.006662 DISP= 0.006668
N= 11 X=1.55 EXCF= 0.001686  CF= 0.001690   EXDIS= 0.006772 DISP= 0.006777
N= 12 X=1.60 EXCF= 0.001660  CF= 0.001664   EXDIS= 0.006880 DISP= 0.006885
N= 13 X=1.65 EXCF= 0.001635  CF= 0.001638   EXDIS= 0.006987 DISP= 0.006991
N= 14 X=1.70 EXCF= 0.001610  CF= 0.001614   EXDIS= 0.007092 DISP= 0.007095
N= 15 X=1.75 EXCF= 0.001587  CF= 0.001591   EXDIS= 0.007195 DISP= 0.007197
N= 16 X=1.80 EXCF= 0.001565  CF= 0.001569   EXDIS= 0.007297 DISP= 0.007298
```

```
N= 17 X=1.85 EXCF= 0.001544  CF= 0.001547  EXDIS= 0.007398 DISP= 0.007398
N= 18 X=1.90 EXCF= 0.001523  CF= 0.001527  EXDIS= 0.007497 DISP= 0.007496
N= 19 X=1.95 EXCF= 0.001504  CF= 0.001507  EXDIS= 0.007595 DISP= 0.007592
N= 20 X=2.00 EXCF= 0.001485  CF= 0.001488  EXDIS= 0.007692 DISP= 0.007687
RMS=  0.895E-03
```

15.10 The c_f and δ^* distributions obtained on running program DOROD for the cases indicated are shown in Figs. 1 and 2. When JMAX =11 or 21 a good agrement with the experimental results is apparent. With JMAX=6 too few points across the boundary layer leads to inaccurate results. The parameter settings for these runs were: NMAX=500, XO = 0.938, XMAX= 3.81, DXCH= 0.2, RATCH= 0.01, DX= 0.001, DXMI= 0.001, DXMA= 0.01, BETA= 0.6 .

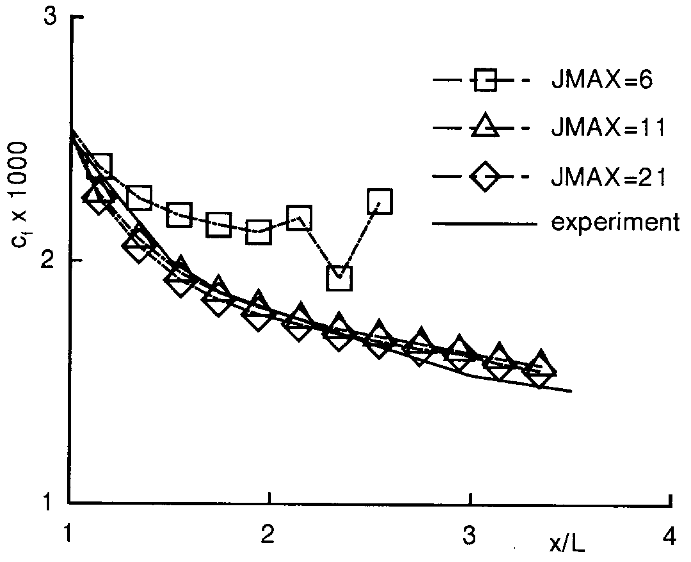

Fig. 1. c_f distribution with a non-uniform y_{max}.

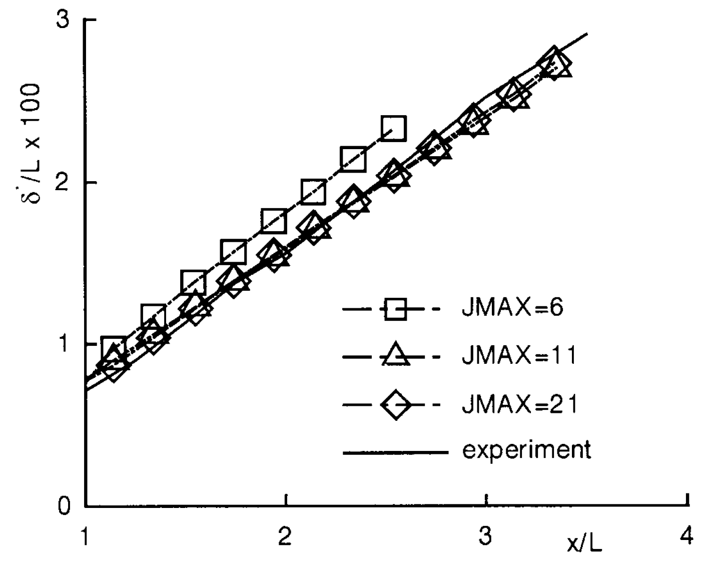

Fig. 2. δ^* distribution with a non-uniform y_{max}.

15.11 On running program DOROD for the cases indicated, it is observed that:

1. the number of stations required to cover a given length of the geometry decreases with increasing $RATCH$, as shown in the following table. Here NS denotes the number of stations required to cover the distance from x=0.94 to x=3.9.

$RATCH$	NS
0.01	407
0.02	196
0.05	74

2. The c_f and δ^* values computed at x≈ 3.0 for various values of $RATCH$ are given below.

$RATCH$	c_f	δ^*
0.01	0.00162	0.0239
0.02	0.00161	0.0239
0.05	0.00159	0.0237

Thus the predictions of C_f and δ^* values are equally accurate for lower values of *RATCH* like 0.01 and 0.02. The number of stations required to cover a given length in the flow direction is seen to be a strong function of *RATCH* which for reasons of economy should be as high as possible.

15.12 The modification to the program consists of printing out the values of u^+ and y^+ from subroutine TURVS.

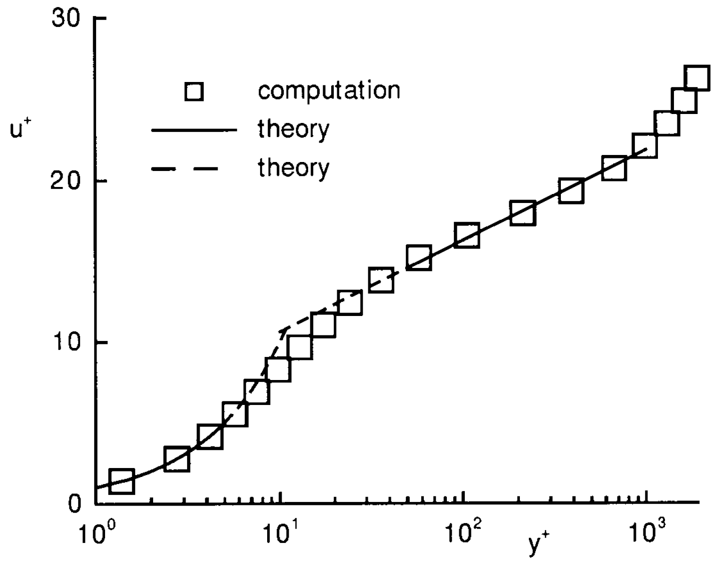

Fig. 1. Velocity profile in wall - coordinates.

The velocity profile in wall coordinates i.e., u^+ vs y^+ (y^+ is plotted on a logarthmic scale) computed with 21 points across the boundary layer, is shown in Fig. 1. Also shown in the figure is the universal curve, which is given by

$$u^+ = y^+, \qquad \text{for} \qquad y^+ \leq 5.0 \ .$$

$$u^+ = \frac{1}{\kappa} \ln y^+ + 5.1 \qquad \text{for} \qquad y^+ \geq 50.0 \ ,$$

and $\kappa = 0.41$.

The agreement between the computed and the theoretical curves is good up to a $y^+ \approx 400$. Strictly the second relation above is valid only upto $y^+ = 400$, beyond which the following expression is to be used.

$$\frac{u_e - \bar{u}}{u_\tau} = -\frac{1}{\kappa} \ln \frac{y}{\delta} + A(1 - \cos(\pi y/\delta)) \ .$$

Where u_e is the velocity at the boundary layer edge, $\bar{u}$ is the mean velocity, δ is the boundary layer thickness and A is a constant depending on the pressure, $p_e(x)$.

15.13 With the use of one-dimensional quadratic elements, the form of (15.62) is unchanged. However the matrices, CC_{kj}, EF_{kj} and AA_{kj} become alternating tridiagonal and pentadiagonal, as is CCC_{kj}, in (15.66). Subroutine *FEPAR* must be modified, either to evaluate *CC*, *EF* and *AA* via algebraic expressions involving *Du* etc., or they must be evaluated numerically. Subroutine *ABCD* has to be changed so that CCC_{kj} is evaluated with the proper alternating tridiagonal / pentadiagonal system without further modification. The rest of program *DOROD* is directly applicable without alteration.

15.14 The c_f and δ^* distributions obtained on running program DOROD for this case are shown in Figs. 1 and 2. Good agreement is seen between the present predictions and the experimental results due to Wieghardt and Tillmann. The parameter settings for this run were: NMAX =300, IMAX =11, XO = 0.187, XMAX= 5.01, DXCH= 0.2, RATCH= 0.02, DX= 0.001, DXMI= 0.001, DXMA= 0.1, BETA= 0.6 .

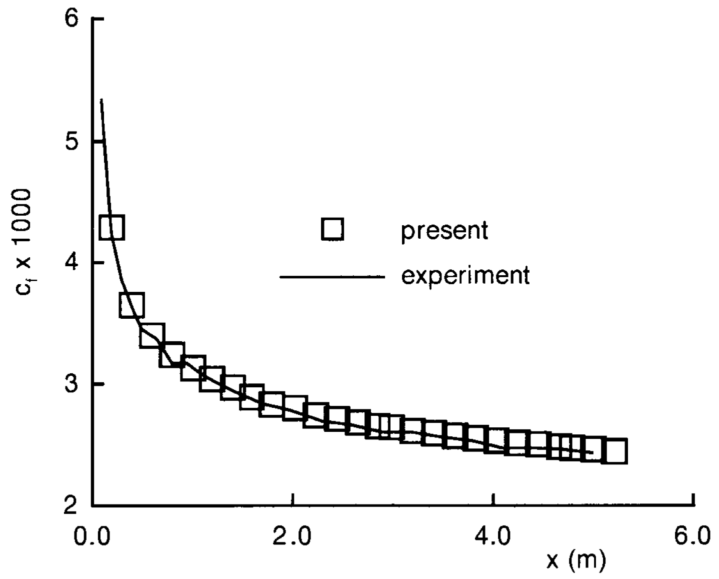

Fig. 1. c_f distribution for turbulent flow over a flat plate.

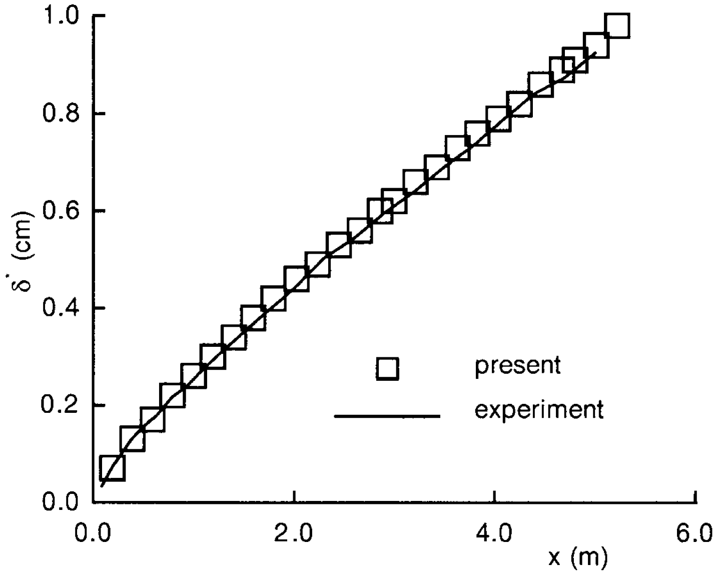

Fig. 2. δ^* distribution for turbulent flow over a flat plate.

15.15 As the first step, we write (15.85) and (15.86) as

$$u\frac{\partial F}{\partial x} + v\frac{\partial F}{\partial y} + w\frac{\partial F}{\partial z} = \frac{1}{\rho}\left(C_1\frac{\partial p_e}{\partial x} + (1-C_1)\frac{\partial p_e}{\partial z}\right) + \nu\frac{\partial^2 F}{\partial y^2}\ , \tag{1}$$

where $F = (u\ ,\ w)^T$, $C_1 = (1\ ,\ 0)^T$.

a) Crank-Nicolson scheme

For the Crank-Nicolson scheme (Fig. 15.23a) the x,y,z derivatives are centred at $n+1/2$,j, $k-1/2$ respectively. Thus (1) is written as

$$\begin{aligned}&(u\frac{\partial F}{\partial x})^{n+1/2}_{j,k} + (v\frac{\partial F}{\partial y})^{n+1}_{j,k} + (w\frac{\partial F}{\partial z})^{n}_{j,k-1/2}\\ &= \frac{1}{\rho}[C_1\frac{\partial p_e}{\partial x} + (1-C_1)\frac{\partial p_e}{\partial z}]^{n+1}_{j,k} + \nu(\frac{\partial^2 F}{\partial y^2})^{n+1}_{j,k}\ .\end{aligned} \tag{2}$$

Writing

$$(u\frac{\partial F}{\partial x})^{n+1/2} \quad \text{as} \quad 0.5(u^n + u^{n+1})(\frac{F^{n+1}-F^n}{\Delta x}) = \frac{u^n}{\Delta x}(F^{n+1}-F^n)\ , \tag{3}$$

and linearising

$$(v\frac{\partial F}{\partial y})^{n+1} \quad \text{as} \quad v^n(\frac{\partial F}{\partial y})^{n+1} + v^{n+1}(\frac{\partial F}{\partial y})^{n} - v^n(\frac{\partial F}{\partial y})^{n} \tag{4}$$

and applying the Crank-Nicolson scheme, we get

$$\begin{aligned}&\frac{u^n_{j,k}}{\Delta x}(F^{n+1}_{j,k} - F^n_{j,k}) + \frac{1}{2}(v\frac{\partial F}{\partial y})^{n+1}_{j,k} + \frac{1}{2}(v\frac{\partial F}{\partial y})^{n}_{j,k} + (w\frac{\partial F}{\partial z})^{n+1}_{j,k-1/2}\\ &= \frac{1}{\rho}[C_1\frac{\partial p_e}{\partial x} + (1-C_1)\frac{\partial p_e}{\partial z}]^{n+1}_{j,k} + \frac{1}{2}\nu(\frac{\partial^2 F}{\partial y^2})^{n}_{j,k} + \frac{1}{2}\nu(\frac{\partial^2 F}{\partial y^2})^{n+1}_{j,k}\ .\end{aligned} \tag{5}$$

Now writing $(w(\partial F/\partial z))^{n+1}_{j,k-1/2}$ as $w^n_{j,k-1/2}(\partial F/\partial z)^{n+1}_{j,k-1/2}$, we have after rearranging,

$$\begin{aligned}&\left(\frac{u^n_{j,k}}{\Delta x} + \frac{1}{2}(w^n_{j,k} + w^n_{j,k+1})\frac{1}{\Delta z} + \frac{0.5\nu}{\Delta y^2}\right)F^{n+1}_{j,k}\\ &+ \left(\frac{0.5v^n_{j,k}}{2\Delta y} - \frac{0.5\nu}{2\Delta y^2}\right)F^{n+1}_{j+1,k}\\ &+ \left(\frac{-0.5v^n_{j,k}}{2\Delta y} - \frac{0.5\nu}{2\Delta y^2}\right)F^{n+1}_{j-1,k}\\ &= \frac{u^n_{j,k}}{\Delta x}F^n_{j,k} - 0.5v^k_{j,k}\frac{F^n_{j+1,k} - F^n_{j-1,k}}{2\Delta y} - \frac{1}{2}(\frac{w^n_{j,k} + w^n_{j,k-1}}{\Delta z})F^{n+1}_{j,k-1}\\ &+ \frac{1}{\rho}[C_1\frac{\partial p_e}{\partial x} + (1-C_1)\frac{\partial p_e}{\partial z}]^{n+1}_{j,k} + \frac{0.5\nu}{\Delta y^2}(F^n_{j+1,k} - 2F^n_{j,k} + F^n_{j-1,k})\ .\end{aligned} \tag{6}$$

b) Krause-zig-zag scheme

The derivation of the tridiagonal equation for the Krause zig-zag scheme follows the same lines as before. But now, following (4), we have, $(w(\partial F/\partial z))^{n+1/2}_{j,k}$ instead of $(w(\partial F/\partial z))^{n+1}_{j,k-1/2}$ and $(w(\partial F/\partial z))^{n+1/2}_{j,k}$ is given by (15.88).

Consequently, the tridiagonal equation is

$$\begin{aligned}&\left(\frac{u^n_{j,k}}{\Delta x} + \frac{w^n_{j,k}}{\Delta z} + \frac{0.5\nu}{\Delta y^2}\right)F^{n+1}_{j,k} + \left(\frac{0.5v^n_{j,k}}{2\Delta y} - \frac{0.5\nu}{2\Delta y^2}\right)F^{n+1}_{j+1,k}\\ &+ \left(\frac{-0.5v^n_{j,k}}{2\Delta y} - \frac{0.5\nu}{2\Delta y^2}\right)F^{n+1}_{j-1,k}\\ &= \frac{u^n_{j,k}}{\Delta x}F^n_{j,k} - 0.5v^k_{j,k}\frac{F^n_{j+1,k} - F^n_{j-1,k}}{2\Delta y}\\ &- 0.5\frac{w^n_{j,k}}{\Delta z}(F^n_{j,k+1} + F^n_{j,k} + F^{n+1}_{j,k-1})\\ &+ \frac{1}{\rho}[C_1\frac{\partial p_e}{\partial x} + (1-C_1)\frac{\partial p_e}{\partial z}]^{n+1}_{j,k} + \frac{0.5\nu}{\Delta y^2}(F^n_{j+1,k} - 2F^n_{j,k} + F^n_{j-1,k})\ .\end{aligned} \tag{7}$$

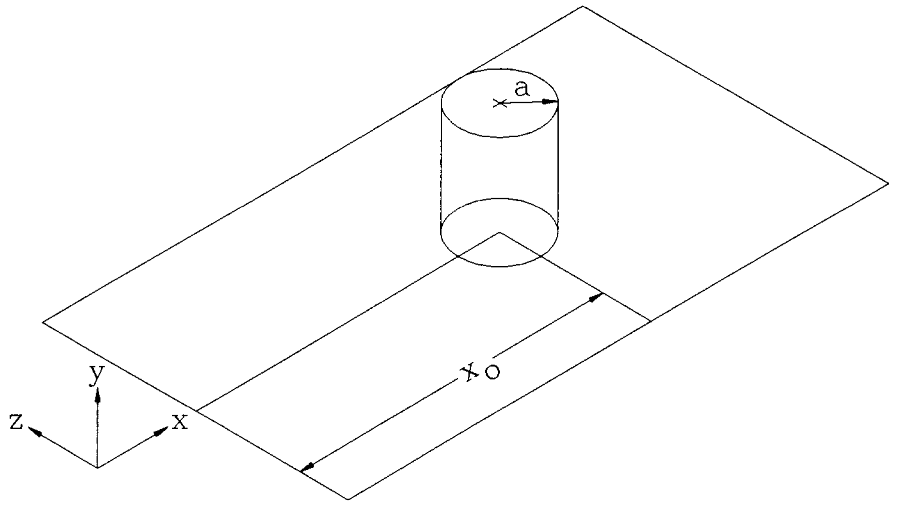

Fig. 1. Geometry of the test case.

15.16 The geometry of the test case is given in Fig. 1.

The physical dimensions are as follows: x_0 = 45.7cm, a = 6.1 cm. The free stream velocity , U_∞=30.5 m/s. The algorithm to compute this flow has been derived in Problem 15.15 and the procedure to implement it is as follows.
1. Generate a grid, which can be uniform in z and y directions. In the x-direction the grid may be uniform or adaptive as in Problem 15.4.
2. Starting from the leading edge of the plate we march in the x direction. At every x station, we sweep first in the y direction and then in the z direction.
3. In performing the sweeps (6) of Problem 15.15 is used. The algorithm has the pressure gradient terms on the right hand side. These are to be evaluated from the equations given for U_e and W_e in the problem definition and the Bernoulli equation.
4. The boundary conditions are :
 y = 0, u = 0 = w = v,
 $y = y_{max}$, u =U_e and w=W_e,
 z = 0, z_{max} : velocity given by flat plate boundary layer solution.
 x = 0, flat plate boundary layer solution.
5. The marching procedure is carried out until axial separation of the flow is encountered. It is expected that this will happen upstream of the cylinder, at y=0. The corresponding axial location provides the downstream limit of the boundary layer calculation.
6. Advantage can be taken of the fact that flow is symmetrical about the centre line on the z- axis. On this line w = 0 and , $(\partial p/\partial z)$ = 0, and these give rise to a singularity in the z-momentum equation. One way to arrive at a non-singular equation is to differentiate the z-momentum equation. This aspect is fully discussed by Cebeci (1975).

CTFD Solutions Manual: Chapter 16

Flows Governed by Reduced Navier-Stokes Equations

16.1 The first step is to nondimensionalise the equations as for (16.1) to (16.4). The continuity equation cannot be simplified further, which implies v is $O(\delta/L)$. All terms on the left-hand side of the x-momentum equation are O(1). On the right-hand side,

$$\frac{1}{Re}\frac{\partial^2 u}{\partial x^2} \text{ is O}\left(\frac{\delta}{L}\right)^3, \quad \frac{1}{Re}\frac{\partial^2 u}{\partial y^2} \text{ and } \frac{\partial}{\partial x}\overline{u'u'} \text{ are O}\left(\frac{\delta}{L}\right), \quad \frac{\partial}{\partial y}\overline{u'v'} \text{ is O(1).}$$

Thus the x-momentum equation becomes,

$$u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + \frac{\partial p}{\partial x} = -\frac{\partial}{\partial y}(\overline{u'v'}).$$

On the left-hand side of the y-momentum equation the convection terms are $O(\delta/L)$. On the right-hand side,

$$\frac{1}{Re}\frac{\partial^2 v}{\partial x^2} \text{ is O}\left(\frac{\delta}{L}\right)^4, \quad \frac{1}{Re}\frac{\partial^2 v}{\partial y^2} \text{ is O}\left(\frac{\delta}{L}\right)^2, \quad \frac{\partial}{\partial x}\overline{u'v'} \text{ is O}\left(\frac{\delta}{L}\right),$$

$$\text{and } \frac{\partial}{\partial y}\overline{v'v'} \text{ is O(1).}$$

This implies that $\partial p/\partial y$ is O(1).

However close to the surface, the normal stress, $-\rho\overline{v'v'}$, is suppressed so that $\partial\overline{v'v'}/\partial y$ is $O(\delta/L)$. Consequently all terms of size $O(\delta/L)$ and bigger are retained. Thus the y-momentum equation becomes,

$$u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + \frac{\partial p}{\partial y} = -\frac{\partial}{\partial y}(\overline{v'v'}).$$

16.2 Retention of all terms in the continuity equation implies v is $O(\delta/L)$. In the x-momentum equation v^2K_{21} is $O(\delta/L)^2$ and $(1/Re)\partial^2u/\partial x^2$ is $O(\delta/L)^3$. These terms are discarded. The terms uvK_{12} and $(1/Re)\partial^2u/\partial y^2$ are $O(\delta/L)$ and are retained. All other terms are 0(1).

In the y-momentum equation, $(1/Re)\partial^2v/\partial x^2$ is $O(\delta/L)^4$ and $(1/Re)\partial^2v/\partial y^2$ is $O(\delta/L)^2$. These terms are discarded. The terms u^2K_{12} and $(1/h_2)\partial p/\partial y$ are

O(1); all other terms are $O(\delta/L)$. The reduced form is based on retaining all terms that are $O(\delta/L)$ or bigger.

16.3 The term, $\exp(i\sigma_y y)$, indicates oscillatory behaviour in the y direction with σ_y real. The character of the solution behaviour in the x direction is of primary interest. The expected behaviour of T(x) for the various cases is as follows.

i) T is oscillatory and growing
ii) T is growing with small oscillations
iii) T is growing slowly with large oscillations
iv) T is growing rapidly with large oscillations
v) T is reducing rapidly with large oscillations

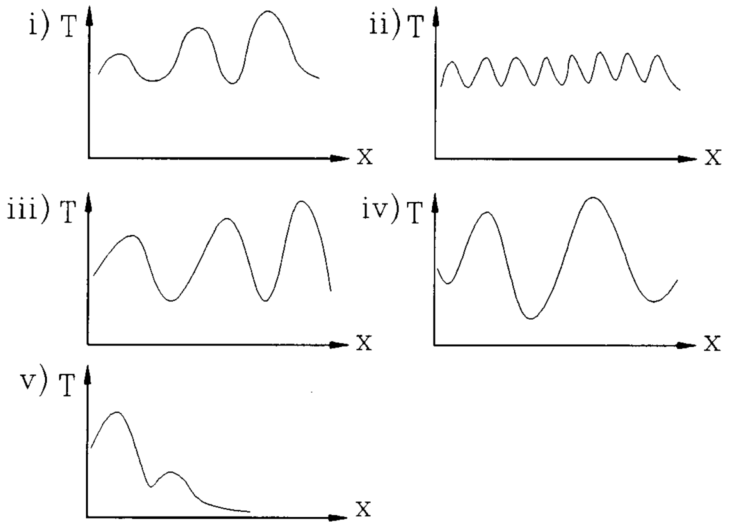

Fig. 1. Behaviour of T(x).

16.4 i) With $\partial p/\partial x = 0$ in (16.1) to (16.3) the symbolic polynomial is

$$\sigma_y^2\left\{i(u\sigma_x + v\sigma_y) + (\sigma_x^2 + \sigma_y^2)/Re\right\} = 0.$$

This has roots

$$\sigma_x = i\sigma_y^2/(uRe) - \sigma_y v/u \quad \text{and} \quad -i\{uRe + \sigma_y^2/(uRe)\} + \sigma_y v/u.$$

Thus the solution will grow in the x direction in an oscillatory manner. This corresponds to the elliptic nature of the governing equations.

ii) With $\partial p/\partial x = 0$ in (16.4) to (16.6) the symbolic polynomial is

$$\sigma_y^2\{i(u\sigma_x + v\sigma_y) + \sigma_y^2/Re\} = 0.$$

Thus $\sigma_x = i\sigma_y^2/(uRe) - \sigma_y\ v/u$.

This produces a damped oscillatory solution in the x direction if u is positive. In this case the equation system is non-elliptic in the x direction.

16.5 For (16.30) to (16.32) with an extra dissipative term, $\epsilon\partial^2\rho/\partial y^2$, on the right-hand side of (16.30), symbolic analysis produces the following equation in place of (16.38),

$$\frac{\sigma_x}{\sigma_y} = -\frac{uv}{(u^2-a^2)} + i\frac{0.5u\sigma_y}{(u^2-a^2)}\{\gamma/(\rho Re)+\epsilon\}$$
$$\pm\frac{a^2}{(u^2-a^2)}\Big[(M^2-1) + (M_x^2-1)\frac{\gamma\epsilon\sigma_y^2}{\rho Re} - 0.25\frac{M_x^2\sigma_y^2}{a^2}\{\gamma/(\rho Re)+\epsilon\}^2$$
$$- i\frac{v\sigma_y}{a^2}\{\gamma/(\rho Re)+\epsilon\}\Big]^{1/2}.$$

Although a precise result is not obvious it might be concluded that a negative value of ϵ will be more stabilising for subsonic flow.

16.6 Results for program THRED for various cases are shown in the table.

DX	DY	RED-FDM RMS error	RED-FEM RMS error	AF-FEM RMS error
0.20	0.20	0.0090	—	0.0030
0.05	0.20	0.0011	0.0035	—
0.05	0.10	0.0011	—	—
0.01	0.20	0.0018	0.0019	—
0.01	0.10	0.0004	—	—

A significant improvement in the accuracy of RED-FDM occurs in refining the grid to DX=0.05 . At this level, the discretisation in x controls the accuracy so that reducing DY to 0.10 makes no change. The relatively poor accuracy of RED-FEM is due to the rapid change in T close to $x = 0$, $y = \pm 1.0$ on a coarse grid. The FEM scheme is sensitive to this. For DX = 0.01 and corresponding refinement of DY to 0.10 is necessary to improve the accuracy.

Based on an approximate operation count analysis, AF-FEM is about forty times more expensive than RED-FDM on the same grid. To achieve comparable accuracy RED-FDM is about ten times more efficient. For the present problem RED-FDM and RED-FEM have a similar operation count. For a three-dimensional duct problem it is expected that ADI-FDM would require a smaller operation count.

16.7 In the radial momentum equation, (16.53), the convective terms are $O(\delta/L)$ and the viscous terms are $O(\delta/L)^2$. For the pressure gradient term, $\partial p^c/\partial r$, to balance the convective terms it is necessary that p^c is $O(\delta/L)^2$.

This implies that $\partial p^c/\partial x$ in the axial momentum equation is $O(\delta/L)^2$ and can be discarded. In contrast the term $dp_{c/l}/dx$ is O(1).

16.8 Since $p_{c/l}$ is obtained from the global conservation of mass, equations (16.51) to (16.53) determine u, v and p^c. Therefore introducing Fourier representations for these three variables produces the following symbolic polynomial,

$$i\sigma_r^2(u\sigma_x + \{v - 1/(rRe)\}\sigma_r - i\sigma_r^2/Re) = 0,$$

$$\text{or} \quad \sigma_x = \{1/(rRe) - v\}\sigma_r/u + i\sigma_r^2/(Re\ u).$$

This will produce a stable marching solution as long as u is positive.

16.9 The discrete form of (16.55) can be written

$$u_j^n \Delta u_j^{n+1}/\Delta x = J_d(u_j^{n+1/2},\ v_j^{n+1/2}, r_j) - \Delta p_{c/l}^{n+1}/\Delta x. \qquad (1)$$

Since J_d depends linearly on $u^{n+1/2}$,

$$J_d(u_j^{n+1/2}) = 0.5\left\{J_d(u_j^n) + J_d(u_j^{n+1})\right\}$$

$$\text{and} \quad J_d(u_j^{n+1}) = J_d(u_j^n) + (\partial J_d/\partial u)\Delta u_j^{n+1}.$$

Thus (1) becomes

$$u_j^n \Delta u_j^{n+1}/\Delta x = J_d(u_j^n) + 0.5(\partial J_d/\partial u)\Delta u_j^{n+1} - \Delta p_{c/l}^{n+1}/\Delta x$$

$$\text{or} \quad \{u_j^n - 0.5\Delta x(\partial J_d/\partial u)\}\Delta u_j^{n+1} = \Delta x\ J_d(u_j^n) - \Delta p_{c/l}^{n+1},$$

which is (16.60).

16.10 Introduction of Fourier series for u, v, w and p in (16.80) to (16.83) produces the following symbolic polynomial,

$$-(\sigma_x^2 + \sigma_y^2 + \sigma_z^2)\{i(u\sigma_x + v\sigma_y + w\sigma_z) + (\sigma_y^2 + \sigma_z^2)/Re\} = 0.$$

The first factor produces a negative imaginary solution for σ_x which implies that the equation system is elliptic and a marching procedure will grow exponentially in the x direction.

If the pressure splitting, (16.84), is exploited and a Fourier series introduced for p^c the resulting symbolic polynomial is

$$-(\sigma_y^2 + \sigma_z^2)\{i(u\sigma_x + v\sigma_y + w\sigma_z) + (\sigma_y^2 + \sigma_z^2)/Re\} = 0.$$

The solution for σ_x comes from the second factor and has no negative imaginary contribution as long as u is positive. Consequently a marching procedure in the x direction is able to produce a stable solution.

16.11 For (16.97) to be true at the discrete level we require

$$\int f_d\ dA + \Delta \bar{f}_d\ A = \int \partial\phi/\partial n\ ds,$$

where $\partial\phi/\partial n$ is either set as a boundary condition or is obtained from the current solution. Thus

$$\Delta \bar{f}_d = \frac{1}{A}\left[\int \partial\phi/\partial n\ ds - \int f_d\ dA\right].$$

16.12 The equation system equivalent to (16.113) to (16.119) can be written

$$\partial u/\partial x + \partial v_\phi/\partial y + \partial w_\phi/\partial z = 0,$$

$$L^c u + \partial p/\partial x - L^d u = 0,$$

$$L^c(\beta v_\phi + v_\psi) + \partial p/\partial y - L^d v_\psi = 0,$$

$$L^c(\beta w_\phi + w_\psi) + \partial p/\partial z - L^d w_\psi = 0,$$

$$\partial v_\psi/\partial y + \partial w_\psi/\partial z = 0,$$

$$\partial v_\phi/\partial z - \partial w_\phi/\partial z = 0,$$

where $L^c u = u\partial u/\partial x + v\partial u/\partial y + w\partial u/\partial z$ and $L^d u = (1/Re)\{\partial^2 u/\partial y^2 + \partial^2 u/\partial z^2\}$, etc.

Introducing Fourier series for u, v_ϕ, v_ψ, w_ϕ, w_ψ and p produces the following symbolic polynomial,

$$(\sigma_y^2 + \sigma_z^2)\Lambda^t\{\beta\sigma_x^2\Lambda^c + (\sigma_y^2 + \sigma_z^2)\Lambda^t\} = 0,$$

where $\Lambda^c = u\sigma_x + v\sigma_y + w\sigma_z$ and $\Lambda^t = \Lambda^c - i(\sigma_y^2 + \sigma_z^2)/Re$.

Setting $\beta = 0$ is equivalent to imposing (16.116). Then σ_x has no negative imaginary value as long as u is positive. However if β is non-zero then σ_x will have a negative imaginary component and a marching solution in the x direction will be unstable.

16.13 The equivalent of (16.126) and (16.127) are

$$w = W^i + \partial\phi/\partial z - \partial\psi/\partial y = 0$$

$$\text{and} \quad \Omega_1 = \partial W^i/\partial y - \partial^2\psi/\partial y^2.$$

Discretising and substituting for $\psi_{j+1,k}$ gives

$$\Omega_{1,j,k} = [\partial W^i/\partial y]_w - 2\psi_{j-1,k}/\Delta y^2 - 2\{W_{j,k}^i + (\phi_{j,k+1} - \phi_{j,k-1})/2\Delta z\}/\Delta y.$$

16.14 Introducing Fourier series for ρ, u and v and substituting for p gives

$$\rho(\rho\Lambda - i\sigma_y^2/Re)\{\Lambda^2 - i\Lambda\sigma_y^2/(\rho Re) - [(\gamma-1)/\gamma]\Lambda\ u\sigma_x - a^2\sigma_x^2/\gamma\} = 0,$$

with $\Lambda = u\sigma_x + v\sigma_y$. Further manipulation gives the required result. If u is positive the first factor does not contain a negative imaginary part. If the viscous term (containing Re) is neglected, the result is

$$\sigma_x/\sigma_y = [(\gamma+1)uv \pm uv\{(\gamma-1)+4\gamma/M_x^2\}^{1/2}]/[2(a^2-u^2)].$$

For $\gamma > 1$, this solution is real; therefore a marching solution in the x direction will be stable.

16.15 Introducing Fourier series for ρ, u and v produces the following symbolic polynomial after some simplification,

$$\rho^2(u\sigma_x+v\sigma_y)[(u\sigma_x/\gamma+v\sigma_y)\{u\sigma_x+v\sigma_y-\omega[(\gamma-1)/\gamma]v\sigma_y\}$$
$$-\omega[(\gamma-1)/\gamma]^2\, uv\sigma_x\sigma_y-(\sigma_x^2+\omega\sigma_y^2)a^2/\gamma]=0.$$

After further simplification this gives the required result. If $v \approx 0$ then the major factor gives

$$\sigma_x = \pm\omega^{1/2}\sigma_y a/(u^2-a^2)^{1/2}.$$

If $u < a$, σ_x will have a negative imaginary solution and exponential growth in x will occur.

16.16 Introducing Fourier series for u, v and p produces the following symbolic polynomial,

$$\{u\sigma_x+v\sigma_y-i\sigma_y^2/Re\}\{\sigma_x^2+i\alpha\sigma_x^2\sigma_t+\sigma_y^2\}=0.$$

From the second factor, $\sigma_t = i(\sigma_x^2+\sigma_y^2)/(\alpha\sigma_x^2)$. Thus exponential growth in time is avoided if α is positive.

16.17 From (16.178) and (16.180)

$$\frac{\partial p}{\partial y}=\frac{1}{Re}\frac{\partial^2 v}{\partial y^2}-\rho u\frac{\partial v}{\partial x}-\rho v\frac{\partial v}{\partial y}. \tag{1}$$

Equation (16.181) can be used to obtain $\partial\rho/\partial x$ and $\partial\rho/\partial y$ which are substituted into (16.178). The result is

$$\rho\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)+\frac{\gamma}{a^2}\left(u\frac{\partial p}{\partial x}+v\frac{\partial p}{\partial y}\right)$$
$$+(\gamma-1)\rho\left[M_x^2\frac{\partial u}{\partial x}+M_xM_y\left(\frac{\partial v}{\partial x}+\frac{\partial u}{\partial y}\right)+M_y^2\frac{\partial v}{\partial y}\right]=0.$$

Substitution of $\partial p/\partial y$ from (1), and rearrangement, gives (16.84).

16.18 The physical grid used for u, v and p is indicated in Fig. 16.22. Equation (16.191) is, in physical coordinates,

$$(u_{j,k-1/2}-u_{j-1,k-1/2})/\Delta x+(v_{j-1/2,k}-v_{j-1/2,k-1})/\Delta y=0.$$

A Taylor series expansion about $(j, k-1/2)$ indicates that the first term is $O(\Delta x)$ and the second term is $O(\Delta x, \Delta y^2)$. The term v is defined at $(j-1/2, k)$ but $u_{j,k-1/2}$ is evaluated as 0.5 $(u_{j,k}+u_{j,k-1})$.

Equation (16.192) is, in physical coordinates,

$$u_{j,k}(u_{j,k}-u_{j-1,k})/\Delta x+v_{j-1/2,k}(u_{j,k+1}-u_{j,k-1})/2\Delta y$$
$$+(p_{j,k}-p_{j-1,k})/\Delta x-(u_{j,k-1}-2u_{j,k}+u_{j,k+1})/(Re\ \Delta y^2)=0.$$

A Taylor series expansion about (j,k) shows that the terms are $O(\Delta x)$, $O(\Delta x, \Delta y^2)$, $O(\Delta x)$ and $O(\Delta y^2)$ respectively.

Equation (16.193) is, in physical coordinates,

$$u_{j,k-1/2}(v_{j-1/2,k-1/2}-v_{j-3/2,k-1/2})/\Delta x$$
$$+\ v_{j-1/2,k-1/2}(v_{j-1/2,k}-v_{j-1/2,k-1})/\Delta y$$
$$+\ (p_{j,k}-p_{j,k-1})/\Delta y=0.$$

A Taylor series expansion about $(j-1, k-1/2)$ indicates that the terms are $O(\Delta x)$, $O(\Delta x, \Delta y^2)$ and $O(\Delta x, \Delta y^2)$ respectively.

16.19 From the definition of the displacement thickness, the loss of mass flow is given by

$$u_e\delta^* = \int_o^\delta (u_e-u)dy.$$

However assuming an injection velocity at the surface, v_i, and applying the continuity equation to a strip of height δ^* implies

$$v_i\ dx=\left\{u_e\delta^*+\frac{\partial}{\partial x}(u_e\delta^*)dx-u_e\delta^*\right\}\quad\text{or}\quad v_i=\frac{d}{dx}(u_e\delta^*).$$

For compressible flow the equivalent expression would be

$$(\rho v)_i=\frac{d}{dx}(\rho_e u_e\delta^*).$$

16.20 Subtracting (16.221) from (16.220) and integrating across the boundary layer gives

$$[G^i-G^v]_{y=0}=\frac{\partial}{\partial x}\int_o^\delta (F^i-F^v)\ dy.$$

Substituting $F^v = F^i + F^b - F^i_{y=0}$ leads to

$$G^i_{y=0}=G^b_{y=0}+\frac{\partial}{\partial x}\int_o^\delta (F^i_{y=0}-F^b)\ dy.$$

For the various components,

$$[\rho v]^i_{y=0} = \frac{\partial}{\partial x}\int_o^\delta \{(\rho u)_e - (\rho u)^b\}\, dy$$

$$[\rho uv]^i_{y=0} = -\tau_{yx}\big|_{y=0} + \frac{\partial}{\partial x}\int_o^\delta \{(\rho uv)_e - (\rho uv)^b\}\, dy$$

$$[(E+p)v]^i_{y=0} = \frac{\partial}{\partial x}\int_o^\delta \{[(E+p)u]_e - [(E+p)u]^b\}\, dy.$$

CTFD Solutions Manual: Chapter 17

Incompressible Viscous Flow

17.1 Consider the square domain shown in Fig. 1.

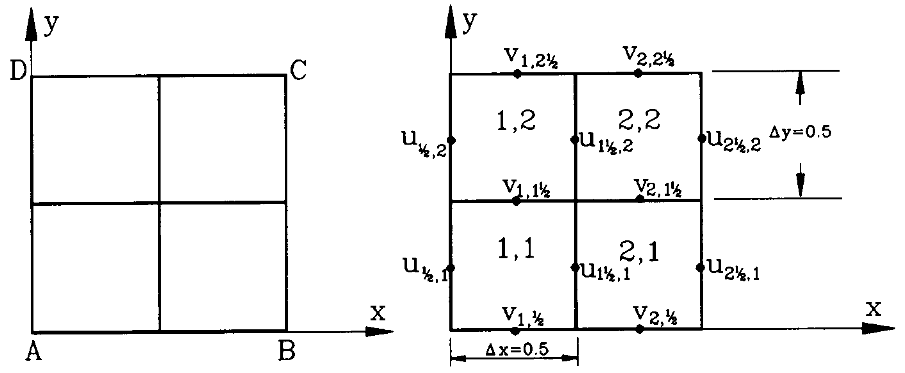

Fig. 1. Typical domain.

From the Stokes theorem we have

$$\begin{aligned}\int\int\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)dx\ dy &= \int\Big[\, u\,\Big]_{AD}^{BC} dy + \Big[\, v\,\Big]_{AB}^{DC} dx \\ &= -\int_{AB} vdx + \int_{BC} udy + \int_{CD} vdx - \int_{DA} udy \\ &= \oint v\cdot n\ ds = 0\ . \end{aligned} \qquad (1)$$

Now divide ABCD into four cells as shown.

$$
\begin{aligned}
&\int\!\!\int\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)dx\ dy\\
&=\int\!\!\int_{1,1}(\ \ )dx\ dy+\int\!\!\int_{1,2}(\ \ )dx\ dy+\int\!\!\int_{2,1}(\ \ )dx\ dy+\int\!\!\int_{2,2}(\ \ )dx\ dy\\
&=\left(\frac{u_{3/2,1}-u_{1/2,1}}{\Delta x}+\frac{v_{1,3/2}-v_{1,1/2}}{\Delta y}\right)\Delta x\ \Delta y\\
&+\left(\frac{u_{3/2,2}-u_{1/2,2}}{\Delta x}+\frac{v_{1,5/2}-v_{1,3/2}}{\Delta y}\right)\Delta x\ \Delta y\\
&+\left(\frac{u_{5/2,1}-u_{3/2,1}}{\Delta x}+\frac{v_{2,3/2}-v_{2,1/2}}{\Delta y}\right)\Delta x\ \Delta y\\
&+\left(\frac{u_{5/2,2}-u_{3/2,2}}{\Delta x}+\frac{v_{2,5/2}-v_{2,3/2}}{\Delta y}\right)\Delta x\ \Delta y\\
&=0\\
&=-v_{1,1/2}\Delta x-v_{2,5/2}\Delta x+u_{5/2,1}\Delta y+u_{5/2,2}\Delta y\\
&\quad+\quad v_{2,5/2}\Delta x+v_{1,5/2}\Delta x-u_{1/2,2}\Delta y-u_{1/2,1}\Delta y\ .
\end{aligned}
\qquad (2)
$$

Thus showing that even the discretised form of the continuity equation satisfies (17.4).

17.2 The discretised form of the continuity equation (17.12), is

$$
D^{n+1}_{j,k}=\frac{1}{\Delta x}\left(u^{n+1}_{j+1/2,k}-u^{n+1}_{j-1/2,k}\right)+\frac{1}{\Delta y}\left(v^{n+1}_{j,k+1/2}-v^{n+1}_{j,k-1/2}\right)=0\ . \qquad (1)
$$

Substituting for $u^{n+1}_{j+1/2,k}$, etc. from (17.8) and (17.10), we have

$$
\begin{aligned}
D^{n+1}_{j,k}=&\frac{1}{\Delta x}\left\{F^n_{j+1/2,k}-\frac{\Delta t}{\Delta x}\left(p^{n+1}_{j+1,k}-p^{n+1}_{j,k}\right)-F^n_{j-1/2,k}\right.\\
&\left.+\frac{\Delta t}{\Delta x}\left(p^{n+1}_{j,k}-p^{n+1}_{j-1,k}\right)\right\}\\
&+\frac{1}{\Delta y}\left\{G^n_{j,k+1/2}-\frac{\Delta t}{\Delta y}\left(p^{n+1}_{j,k+1}-p^{n+1}_{j,k}\right)-G^n_{j,k-1/2}\right.\\
&\left.+\frac{\Delta t}{\Delta y}\left(p^{n+1}_{j,k}-p^{n+1}_{j,k-1}\right)\right\}=0\ .
\end{aligned}
\qquad (2)
$$

Simplifying (2),

$$
\begin{aligned}
&\left[\frac{p_{j+1,k}-2p_{j,k}+p_{j-1,k}}{\Delta x^2}+\frac{p_{j,k+1}-2p_{j,k}+p_{j,k-1}}{\Delta y^2}\right]^{n+1}\\
&=\frac{1}{\Delta t}\left[\frac{F_{j+1/2,k}-F_{j-1/2,k}}{\Delta x}+\frac{G_{j,k+1/2}-G_{j,k-1/2}}{\Delta y}\right]^{n},
\end{aligned}
\qquad (3)
$$

as required.

In order to derive (17.14), we have to substitute for the terms on the right-hand side of (3) from (17.9 and 17.11). For example, the leading terms in each expression give

$$
D^n_{j,k}=(u^n_{j+1/2,k}-u^n_{j-1/2,k})/\Delta x+(v^n_{j,k+1/2}-v^n_{j,k-1/2}/\Delta y\ .
$$

After additional manipulation we get

$$
RHS_{(3)}=\frac{D^n_{j,k}}{\Delta t}+\frac{1}{Re}(L_{xx}+L_{yy})D^n_{j,k}-L_{xx}\ u^2_{j,k}-L_{yy}\ v^2_{j,k}-2L_{xy}(uv)_{j,k}\ ,
$$

as required.

17.3 Substituting for $p_{1,2}-p_{0,2}$ from (17.19) into (17.18), we have

$$
\begin{aligned}
&-\left[\frac{-p_{1,2}+p_{2,2}}{\Delta x^2}+\frac{p_{1,1}-2p_{1,2}+p_{1,3}}{\Delta y^2}\right]^{n+1}\\
&=\frac{F^n_{1/2,2}-u^{n+1}_{1/2,2}}{\Delta x\ \Delta t}+\frac{1}{\Delta t}\left[\frac{F^n_{3/2,2}-F^n_{1/2,2}}{\Delta x}+\frac{G^n_{1,5/2}-G^n_{1,3/2}}{\Delta y}\right]\\
&=\frac{F^n_{3/2,2}-u^{n+1}_{1/2,2}}{\Delta x\Delta t}+\frac{G^n_{1,5/2}-G^n_{1,3/2}}{\Delta y\ \Delta t}\ ,
\end{aligned}
$$

which is independent of $F^n_{1/2,2}$. The argument following (17.18) and (17.19) is thus confirmed.

17.4 If u is set equal to a constant, scheme (17.31) will be

$$
\frac{T^{n+1}_j-T^n_j}{\Delta t}+\frac{u}{\Delta x}(T_{j+1/2}-T_{j-1/2})^n-\frac{\alpha}{\Delta x^2}(T_{j+1}-2T_j+T_{j-1})^n=0\ . \qquad (1)
$$

Substituting for $T_{j-1/2}$ and $T_{j+1/2}$ from (17.32 and 17.33),

$$
\begin{aligned}
\frac{u}{\Delta x}(T_{j+1/2}-T_{j-1/2})^n=&\frac{u}{\Delta x}\left(\frac{T_{j+1}-T_{j-1}}{2}\right)+\\
&\frac{q}{3\Delta x}(T_{j-2}-3T_{j-1}+3T_j-T_{j+1})\ ,
\end{aligned}
\qquad (2)
$$

so that scheme (1) now becomes,

$$
\begin{aligned}
&\frac{T^{n+1}_j-T^n_j}{\Delta t}+\frac{u}{2\Delta x}(T_{j+1}-T_{j-1})^n+\\
&\frac{q}{3\Delta x}(T_{j-2}-3T_{j-1}+3T_j-T_{j+1})^n-\frac{\alpha}{\Delta x^2}(T_{j+1}-2T_j+T_{j-1})^n=0\ .
\end{aligned}
\qquad (3)
$$

Scheme (9.71) is given by (written here as a two-level explicit scheme)

$$
\frac{T^{n+1}_j-T^n_j}{\Delta t}+(uL^{(4)}_x-\alpha L_{xx})T^n_j=0\ . \qquad (4)
$$

Substituting for $L^{(4)}_x$ from (9.72), we find that schemes (3) and (9.71) are equivalent.

17.5 First, we will show that (17.40) follows from (17.39). This can be done by replacing the u^* and v^* terms in the equation for RHS^B from the CPSM form of (17.23). Writing the equation for $u^*_{1,k}$ we have from (17.23)

$$\begin{aligned} u^*_{1,k} &= u^{n+1}_{1,k} + \Delta t\ \nabla^{(x)}_d\ p^{n+1}_{1,k} \\ &= u^{n+1}_{1,k} + \Delta t\Big(\frac{\partial p}{\partial x}\Big)^{n+1}_{1,k} \\ &= u^{n+1}_{1,k} + \Delta t\Big(\sum_{l=1}^{N_x+1} G^{x(1)}_{1,l}\, p_{l,k}\Big)\ . \end{aligned} \qquad (1)$$

Similarly

$$u^*_{N_x+1,k} = u^{n+1}_{N_x+1,k} + \Delta t \sum_{l=1}^{N_x+1} G^{x(1)}_{N_x+1,l}\, p_{l,k}\ , \qquad (2)$$

$$v^*_{j,1} = v^{n+1}_{j,1} + \Delta t \sum_{m=1}^{N_y+1} G^{y(1)}_{1,m}\, p_{j,m}\ , \qquad (3)$$

$$v^*_{j,N_y+1} = v^{n+1}_{j,N_y+1} + \Delta t \sum_{m=1}^{N_y+1} G^{y(1)}_{N_y+1,m}\, p_{j,m}\ . \qquad (4)$$

Substituting (1), (2), (3) and (4) into the equation following (17.39) we obtain the equation for RHS^B in the form (17.40). Following a similar procedure one can show that (17.43) and (17.44) follow from (17.41) and (17.42).

17.6 The Jacobians $\underline{\mathbf{A}}$ and $\underline{\mathbf{B}}$ in (17.48) are defined as,

$$\underline{\mathbf{A}} = \frac{\partial \mathbf{F}}{\partial \mathbf{q}},\quad \underline{\mathbf{B}} = \frac{\partial \mathbf{G}}{\partial \mathbf{q}},\quad \text{where}\quad \mathbf{q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} p \\ u \\ v \end{bmatrix}\ ,$$

$$\mathbf{F} = \begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix} = \begin{bmatrix} a^2 u \\ u^2 + p \\ uv \end{bmatrix} \quad \text{and} \quad \mathbf{G} = \begin{bmatrix} G_1 \\ G_2 \\ G_3 \end{bmatrix} = \begin{bmatrix} a^2 v \\ uv \\ v^2 + p \end{bmatrix}\ ,$$

$$\underline{\mathbf{A}} = \frac{\partial \mathbf{F}}{\partial \mathbf{q}} = \begin{bmatrix} \partial F_1/\partial q_1 & \partial F_1/\partial q_2 & \partial F_1/\partial q_3 \\ \partial F_2/\partial q_1 & \partial F_2/\partial q_2 & \partial F_2/\partial q_3 \\ \partial F_3/\partial q_1 & \partial F_3/\partial q_2 & \partial F_3/\partial q_3 \end{bmatrix} = \begin{bmatrix} 0 & a^2 & 0 \\ 1 & 2u & 0 \\ 0 & v & u \end{bmatrix}\ ,$$

as required. The expression for $\underline{\mathbf{B}}$ is similarly derived.

Substituting for various terms it can easily be shown that
$\mathbf{F} = \underline{\mathbf{A}}\mathbf{q} - u\underline{\mathbf{D}}\mathbf{q}$, $\quad \mathbf{G} = \underline{\mathbf{B}}\mathbf{q} - v\underline{\mathbf{D}}\mathbf{q}$.
For example, $F_2 = 1 \cdot p + 2u \cdot u + 0 \cdot v - u \cdot 1 \cdot u = u^2 + p$.

17.7 The term δp in (17.27) is equivalent to δp in (17.77) and to $-\phi/\Delta t$ in (17.61). In both (17.27) and in the SIMPLE algorithm, δp is evaluated as an adjustment to the momentum equation such that continuity will be satisfied. However ϕ is calculated directly to satisfy continuity and used to adjust the pressure field such that the momentum equation is approximately satisfied.

17.8 a) The expressions for $a^u_{j,k}$ and a^u_{nb} may be derived as follows. Starting from (17.68) and substituting for the terms $F^{(1)}_{j+1/2,k}$, $F^{(1)}_{j-1/2,k}$, $G^{(1)}_{j,k+1/2}$ and $G^{(1)}_{j,k-1/2}$, from the relations prior to (17.69), we have

$$\begin{aligned} &\Big(\frac{\Delta x\ \Delta y}{\Delta t}\Big)(u^{n+1}_{j,k} - u^n_{j,k}) + \\ &\quad \Delta y\Big\{0.25(u_{j,k} + u_{j+1,k})^2 - \frac{1}{Re}\frac{u_{j+1,k} - u_{j,k}}{\Delta x} \\ &\quad - 0.25(u_{j,k} + u_{j-1,k})^2 + \frac{1}{Re}\frac{u_{j,k} - u_{j-1,k}}{\Delta x}\Big\} \\ &\quad + \Delta x\Big\{0.25(v_{j,k} + v_{j+1,k})(u_{j,k} + u_{j,k+1}) - \frac{1}{Re}\frac{u_{j,k+1} - u_{j,k}}{\Delta y} \\ &\quad - 0.25(v_{j,k-1} + v_{j+1,k-1})(u_{j,k} + u_{j,k-1}) + \frac{1}{Re}\frac{u_{j,k} - u_{j,k-1}}{\Delta y}\Big\} \\ &\quad + \Delta y(p^{n+1}_{j+1,k} - p^{n+1}_{j,k}) = 0\ . \end{aligned}$$

Expanding some of the terms and linearising terms like $(u^2)^{n+1}$ as $u^n\ u^{n+1}$ we obtain (17.69) with the following expressions

$$\begin{aligned} a^u_{j,k} =& \frac{\Delta x\ \Delta y}{\Delta t} + 0.25\ \Delta y(u_{j+1,k} - u_{j-1,k}) \\ &+ 0.25\ \Delta x(-v_{j,k-1} - v_{j+1,k-1}) + \frac{2}{Re}\Big(\frac{\Delta y}{\Delta x} + \frac{\Delta x}{\Delta y}\Big)\ , \\ a^u_{j+1,k} =&\ 0.25\ \Delta y(u^n_{j+1,k} + u^n_{j,k}) - \frac{1}{Re}\frac{\Delta y}{\Delta x}\ , \\ a^u_{j-1,k} =& -0.25\ \Delta y(u^n_{j-1,k} + u^n_{j,k}) - \frac{1}{Re}\frac{\Delta y}{\Delta x}\ , \\ a^u_{j,k+1} =&\ 0.25\ \Delta y(v_{j,k} + v_{j,k+1}) - \frac{1}{Re}\frac{\Delta x}{\Delta y}\ , \\ a^u_{j,k-1} =&\ 0.25\ \Delta y(v_{j,k-1} + v_{j,k}) - \frac{1}{Re}\frac{\Delta x}{\Delta y}\ , \\ b^u =& -\frac{\Delta x \Delta y}{\Delta t}u^n_{j,k}\ . \end{aligned}$$

b) Expressions for $a^p_{j,k}$ and a^p_{nb} in (17.77) are derived as follows.

Introducing the velocity correction, $u^c_{j,k}$, in the form

$$u^{n+1}_{j,k} = u^*_{j,k} + u^c_{j,k} \qquad (1)$$

and substituting the above equation in (17.67) we have,

$$\begin{aligned}&\Delta y(u^*_{j,k} + u^c_{j,k} - u^*_{j-1,k} - u^c_{j-1,k})\\&\quad + \Delta x(v^*_{j,k} + v^c_{j,k} - v^*_{j,k-1} - v^c_{j,k-1}) = 0\ .\end{aligned} \tag{2}$$

From (17.75) we have,

$$u^c_{j,k} = d^{(x)}_{j,k}(\delta p_{j,k} - \delta p_{j+1,k}) \tag{3}$$

and as a consequence (2) becomes

$$\begin{aligned}&\Delta y\Big\{ d^{(x)}_{j,k}(\delta p_{j,k} - \delta p_{j+1,k}) - d^{(x)}_{j-1,k}(\delta p_{j-1,k} - \delta p_{j,k})\Big\}\\&\quad + \Delta x\Big\{ d^{(y)}_{j,k}(\delta p_{j,k} - \delta p_{j,k+1}) - d^{(y)}_{j,k-1}(\delta p_{j,k-1} - \delta p_{j,k})\Big\}\\&\quad = -\Delta y(u^*_{j,k} - u^*_{j-1,k}) - \Delta x(v^*_{j,k} - v^*_{j,k-1})\ ,\end{aligned} \tag{4}$$

which can be written as (17.77) with

$$\begin{aligned}&a^p_{j,k} = \Delta y(d^{(x)}_{j,k} + d^{(x)}_{j-1,k}) + \Delta x(d^{(y)}_{j,k} + d^{(y)}_{j,k-1})\\&a_{j+1,k} = \Delta y d^{(x)}_{j,k}, \qquad a_{j-1,k} = \Delta y d^{(x)}_{j-1,k},\\&a_{j,k+1} = \Delta x d^{(y)}_{j,k} \qquad a_{j,k-1} = \Delta x d^{(y)}_{j,k-1}\\&\text{and}\quad b^p = -(u^*_{j,k} - u^*_{j-1,k})\Delta y - (v^*_{j,k} - v^*_{j,k-1})\Delta x\ .\end{aligned} \tag{5}$$

To show that (17.77) is a discrete Poisson equation, consider (4) divided by $\Delta x\Delta y$. Let $d^{(x)}_{j-1,k} = d^{(x)}_{j,k} = k/\Delta x$ and $d^{(y)}_{j,k-1} = d^{(y)}_{j,k} = k/\Delta y$. Then (4) reduces to

$$\begin{aligned}&- k(\delta p_{j+1,k} - 2\delta p_{j,k} + \delta p_{j-1,k})/\Delta x^2\\&- k(\delta p_{j,k+1} - 2\delta p_{j,k} + \delta p_{j,k-1})/\Delta y^2\\&= -(u^*_{j,k} - u^*_{j-1,k})/\Delta x - (v^*_{j,k} - v^*_{j,k-1})/\Delta y\ .\end{aligned} \tag{6}$$

The LHS of (6) is a discretised form of

$$-k\{\frac{\partial^2}{\partial x^2}(\delta p_{j,k}) + \frac{\partial^2}{\partial y^2}(\delta p_{j,k})\}\ . \tag{7}$$

This shows that (17.77) is a discretised form of the Poisson equation.

17.9 Application of the three-point upwind formula (9.53) to discretise the convective terms of (17.2) is as follows. We have (assuming $q = 1.5$ and a positive u-velocity)

$$\begin{aligned}&(\frac{\Delta x\Delta y}{\Delta t})(u^{n+1}_{j,k} - u^n_{j,k}) + \frac{1}{2}(u^2_{j-2,k} - 4u^2_{j-1,k} + 3u^2_{j,k})\Delta y\\&+ \frac{1}{2}((uv)_{j,k-2} - 4(uv)_{j,k-1} + 3(uv)_{j,k})\Delta x\\&- \frac{\Delta y}{Re\Delta x}(u_{j+1,k} - 2u_{j,k} + u_{j-1,k})\\&- \frac{\Delta x}{Re\Delta y}(u_{j,k+1} - 2u_{j,k} + u_{j,k-1}) + (p_{j+1,k} - p_{j,k})\Delta y = 0\ .\end{aligned} \tag{1}$$

As a consequence (17.69) changes to

$$\begin{aligned}&(\frac{\Delta x\Delta y}{\Delta t} + a^u_{j,k})u^{n+1}_{j,k} + \sum a^u_{nb}u^{n+1}_{nb} + a^u_{j-2,k}u^{n+1}_{j-2,k} + a^v_{j,k-2}u^{n+1}_{j,k-2}\\&+ b^u + \Delta y(p^{n+1}_{j+1,k} - p^{n+1}_{j,k}) = 0\ .\end{aligned} \tag{2}$$

The expressions for $a^u_{j,k}$, a^u_{nb} will now be different from those derived in Problem 17.8. In addition, the terms, $a^u_{j-2,k}$ and $a^v_{j,k-2}$ will need to be evaluated using $u^n_{j-2,k}$ and $v^n_{j,k-2}$ if the same implicit algorithm is to be used . Thus they should be combined with b^n. Also if $u_{j,k}$ or $v_{j,k}$ are negative contributions will be obtained from nodes (j+2,k) and (j,k+2).

17.10 Equation (17.98) is given by

$$\begin{aligned}\frac{\partial(u\zeta)}{\partial x} \approx\ & \mu_x L^+_x(u\zeta)^{n+1}_{j,k} + (1 - \mu_x)L^-_x(u\zeta)^{n+1}_{j,k}\\&+ 0.5\Delta x(1 - 2\mu_x)L_{xx}(u\zeta)^n_{j,k}\ .\end{aligned} \tag{1}$$

At steady state $(u\zeta)^{n+1}_{j,k} = (u\zeta)^n_{j,k}$, etc.

Hence

$$\begin{aligned}\frac{\partial(u\zeta)}{\partial x} &\approx \mu_x(L^+_x - L^-_x)(u\zeta)^n_{j,k} + L^-_x(u\zeta)^n_{j,k}\\&\qquad + 0.5\Delta x L_{xx}(u\zeta)^n_{j,k} - \Delta x\mu_x L_{xx}(u\zeta)^n_{j,k}\\&\approx \mu_x\Delta x L_{xx}(u\zeta)^n_{j,k} + L^-_x(u\zeta)^n_{j,k}\\&\qquad + 0.5\Delta x L_{xx}(u\zeta)^n_{j,k} - \Delta x\mu_x L_{xx}(u\zeta)^n_{j,k}\\&\approx \frac{1}{\Delta x}\Big[(u\zeta)^n_{j,k} - (u\zeta)^n_{j-1,k}\\&\qquad + 0.5(u\zeta)^n_{j-1,k} - (u\zeta)^n_{j,k} + 0.5(u\zeta)^n_{j+1,k}\Big]\\&\approx \frac{0.5}{\Delta x}\Big[(u\zeta)^n_{j+1,k} - (u\zeta)^n_{j-1,k}\Big]\ ,\end{aligned} \tag{2}$$

which is a three-point centred difference scheme.

17.11 The first-order boundary condition for the vorticity equation is derived as follows (with reference to Fig. 17.12). For the Poisson equation for the stream function, the boundary condition on AD is $\psi = 0$ and for the vorticity equation it is $\partial\psi/\partial y = g$. In Fig. 17.12, $g_{AD} = 1$.

Carrying out a Taylor series expansion of the stream function about a grid point (j, k) on AD gives

$$\begin{aligned}\psi_{j,k-1} &= \psi_{j,k} - \Delta y\left(\frac{\partial\psi}{\partial y}\right)_{j,k} + \frac{\Delta y^2}{2}\left(\frac{\partial^2\psi}{\partial y^2}\right)_{j,k} + O(H)\\ &= \psi_{j,k} - \Delta y g + \frac{\Delta y^2}{2}\left(\frac{\partial^2\psi}{\partial y^2}\right)_{j,k} + O(H)\ .\end{aligned} \tag{1}$$

From (17.92), we have

$$\frac{\partial^2\psi}{\partial x^2} + \frac{\partial^2\psi}{\partial y^2} = \zeta\ . \tag{2}$$

Since on AD, $\psi = 0$, we have $\partial^2\psi/\partial y^2 = \zeta$.

Accordingly (1) reduces to

$$\begin{aligned}\psi_{j,k-1} &= 0 - \Delta y g + \frac{\Delta y^2}{2}(\zeta)\\ \text{or}\quad \zeta &= \frac{2}{\Delta y^2}(\psi_{j,k-1} + \Delta y g) + O(H)\ .\end{aligned} \tag{3}$$

The corresponding second-order boundary condition is derived as follows.

As suggested in the text, discretising the equation $\partial^2\psi/\partial y^2 = \zeta$ gives

$$\zeta_{j,k} = \frac{\psi_{j,k-1} - 2\psi_{j,k} + \psi_{j,k+1}}{\Delta y^2} + O(H)\ . \tag{4}$$

But to use this equation one needs $\psi_{j,k+1}$ which corresponds to a point outside the computational domain. It is possible to eliminate this term through the formula $g_{j,k} = (\partial\psi/\partial y)_{j,k}$. Employing a third-order accurate expression for $(\partial\psi/\partial y)$, we have

$$g_{j,k} = (\partial\psi/\partial y)_{j,k} = (\psi_{j,k-2} - 6\psi_{j,k-1} + 3\psi_{j,k} + 2\psi_{j,k+1})/6\Delta y\ +\ O(H)\ .$$

Thus,

$$\psi_{j,k+1} = \frac{6g_{j,k}\Delta y - \psi_{j,k-2} + 6\psi_{j,k-1} - 3\psi_{j,k}}{2} + O(H)\ . \tag{5}$$

Substituting (5) in (4) the expression for the second-order boundary condition, i.e. (17.107) in the text, is obtained.

17.12 Consider Fig. 1. For a linear interpolation we have,

in elements A and D:

$$\phi^{(x)}_{j-1} = 0.5(1-\xi),\quad \phi^{(x)}_{j} = 0.5(1+\xi)\quad \text{and}\quad \xi = 2(x - x_{j-\frac{1}{2}})/\Delta x.$$

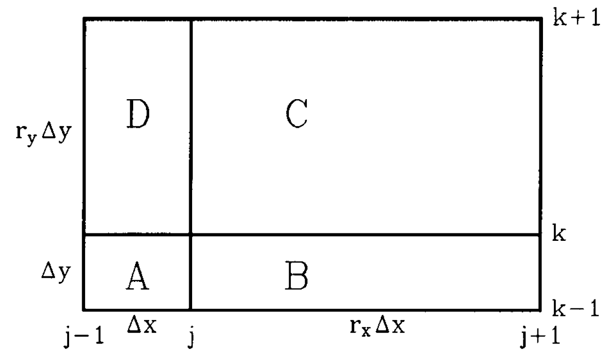

Fig. 1. Linear elements.

In elements B and C:

$$\phi^{(x)}_{j} = 0.5(1-\xi),\quad \phi^{(x)}_{j+1} = 0.5(1+\xi)\quad \text{and}\quad \xi = 2(x - x_{j+\frac{1}{2}})/(r_x\Delta x)\ .$$

For a non-uniform grid (Appendix A.2, Vol. 1), the directional mass operator is

$$M_{xi} = \frac{2}{(1+r_x)\Delta x}\sum_e \int \phi^{(x)}_m \phi^{(x)}_i dx\ .$$

Thus,

$$M_{x1} = \frac{2}{(1+r_x)\Delta x}\cdot\frac{\Delta x}{2}\int_{-1}^{1} 0.25(1-\xi^2)d\xi = \frac{2}{(1+r_x)}\cdot\frac{1}{6}\ ,$$

$$\begin{aligned}M_{x2} &= \frac{2}{(1+r_x)\Delta x}\left[\frac{\Delta x}{2}\int_{-1}^{1} 0.25(1+\xi)^2 d\xi + \frac{r_x\Delta x}{2}\int_{-1}^{1} 0.25(1-\xi)^2 d\xi\right]\\ &= \frac{2}{(1+r_x)}\cdot\frac{(1+r_x)}{3}\ ,\end{aligned}$$

$$M_{x3} = \frac{2}{(1+r_x)\Delta x}\cdot\frac{r_x\Delta x}{2}\int_{-1}^{1} 0.25(1-\xi^2)d\xi = \frac{2}{(1+r_x)}\cdot\frac{r_x}{6}\ .$$

The directional first difference operator is

$$L_{xi} = \frac{2}{(1+r_x)\Delta x}\sum_e \int \phi^{(x)}_m (d\phi^{(x)}_i/dx)dx\ .$$

Thus,

$$L_{x_1} = \frac{2}{(1+r_x)\Delta x}\int_{-1}^{1} -0.25(1+\xi)d\xi = \frac{2}{(1+r_x)}\left(-\frac{0.5}{\Delta x}\right) ,$$

$$L_{x_2} = \frac{2}{(1+r_x)\Delta x}\left[\int_{-1}^{1} +0.25(1+\xi)d\xi - \int_{-1}^{1} 0.25(1-\xi)d\xi\right] = 0 ,$$

$$L_{x_3} = \frac{2}{(1+r_x)\Delta x}\int_{-1}^{1} 0.25(1+\xi)d\xi = \frac{2}{(1+r_x)}\left(\frac{0.5}{\Delta x}\right) .$$

The directional second difference operator is

$$L_{xxi} = \frac{-2}{(1+r_x)\Delta x}\sum_e \int (d\phi_m^{(x)}/dx)\cdot(d\phi_i^{(x)}/dx)dx .$$

Thus,

$$L_{xx1} = \frac{-2}{(1+r_x)\Delta x}\cdot\frac{2}{\Delta x}\int_{-1}^{1} -0.25\ d\xi = \frac{2}{(1+r_x)}\left(\frac{1}{\Delta x^2}\right) ,$$

$$\begin{aligned} L_{xx2} =& \frac{-2}{(1+r_x)\Delta x}\left[\frac{2}{\Delta x}\int_{-1}^{1} 0.25\ d\xi\ +\ \frac{2}{r_x\Delta x}\int_{-1}^{1} 0.25\ d\xi\right] \\ =& \frac{2}{(1+r_x)}\left\{-\frac{(1+1/r_x)}{\Delta x^2}\right\} ,\end{aligned}$$

$$L_{xx3} = \frac{-2}{(1+r_x)\Delta x}\cdot\frac{2}{r_x\Delta x}\int_{-1}^{1} -0.25\ d\xi = \frac{2}{(1+r_x)}\left(\frac{1}{r_x\Delta x^2}\right) ,$$

and similarly for M_y, L_y and L_{yy}.

17.13 Introduction of the three-level time discretisation into (17.114) may be carried out as Sects. 8.3.2 and 9.5.1. As a result we get

$$\begin{aligned} M_x\otimes M_y&\left\{\frac{(1+\gamma)\Delta\zeta^{n+1}-\gamma\Delta\zeta^n}{\Delta t}\right\} \\ &= \beta\ RHS^{n+1} + (1-\beta)\ RHS^n ,\end{aligned} \tag{1}$$

where

$$RHS = \frac{1}{Re}\left\{M_y\otimes L_{xx} + M_x\otimes L_{yy}\right\}\zeta - M_y\otimes L_x u\zeta - M_x\otimes L_y v\zeta . \tag{2}$$

We find that the only term to be linearised in (1) is RHS^{n+1}. Now performing a Taylor series expansion of RHS^{n+1} (which is a function of ζ, u and v) we have

$$\begin{aligned} RHS^{n+1} &= RHS^n \\ &+ \left[\frac{\partial}{\partial\zeta}RHS\frac{\Delta\zeta}{\Delta t}^{n+1} + \frac{\partial RHS}{\partial u}\frac{\Delta u^{n+1}}{\Delta t} + \frac{\partial RHS}{\partial v}\frac{\Delta v^{n+1}}{\Delta t}\right]\Delta t .\end{aligned} \tag{3}$$

Substituting (3) in (1), and simplifying we have

$$\begin{aligned} (1+\gamma)&\left[M_x\otimes M_y - \frac{\Delta t\beta}{1+\gamma}\frac{\partial}{\partial\zeta}(RHS)\right]\Delta\zeta^{n+1} = \\ &\Delta t\ RHS^{n,\beta} + M_x\otimes M_y\ \gamma\Delta\zeta^n ,\end{aligned} \tag{4}$$

where

$$\begin{aligned} RHS^{n,\beta} =& \frac{1}{Re}\left\{M_y\otimes L_{xx} + M_x\otimes L_{yy}\right\}\zeta \\ &- M_y\otimes L_x\zeta(u^n+\beta\Delta u^{n+1}) - M_x\otimes L_y\zeta(v^n+\beta\Delta v^{n+1}) .\end{aligned} \tag{5}$$

To demonstrate that (17.120) and (17.121) are consistent with (17.119), substitute for $\Delta\zeta^*$ in (17.120) from (17.121) and multiply the terms on the LHS. We see that the RHS of the resulting equation and of (17.120) are the same. But the LHS of the equation has the form:

$$LHS_{(17.119)} + \Delta t^2\left(\frac{\beta}{1+\gamma}\right)^2\left(\frac{1}{Re}L_{xx} - L_x u\right)\left(\frac{1}{Re}L_{yy} - L_y v\right)\Delta\zeta^{n+1} .$$

This demonstrates that the (17.120) and (17.121) are consistent with (17.119) to $O(\Delta t^2)$.

17.14 To apply the one-dimensional Galerkin formulation to (17.91), the interpolating functions given by (5.70) are used. The procedure follows closely (5.69 to 5.82), and is not repeated here.

A Taylor series analysis of (17.125) is carried out as follows.

The equation (17.125) given by

$$M_y u = L_y\psi, \qquad \text{i.e. on a uniform grid,} \tag{1}$$

$$\frac{1}{6}u_{j,k-1} + \frac{2}{3}u_{j,k} + \frac{1}{6}u_{j,k+1} = \frac{\psi_{j,k+1}-\psi_{j,k-1}}{2\Delta y} . \tag{2}$$

Expanding the terms as a Taylor series about (j,k), and simplifying, we have

$$u - \psi_y = \frac{\Delta y^2}{6}\psi_{y^3} - \frac{\Delta y^2}{6}u_{yy} - \frac{\Delta y^4}{72}u_{y^4} + O(H) . \tag{3}$$

Substituting for ψ_{y^3} from (17.95) (i.e. $u = \psi_y$), we have

$$u - \psi_y = -\frac{\Delta y^4}{72} u_{y^4} + O(H) \ ,$$

thus showing that the discretisation in (17.125) is fourth-order accurate.

On a non-uniform grid with $y_{k+1} - y_k = r_y(y_k - y_{k-1})$ it can be shown that the leading term in the truncation error is $-\frac{2}{3}(r_y - 1)\Delta y u_y$ and hence the scheme reduces to a first-order accurate one.

17.15 The expression for $g(u)$ on the surface FE in Fig. 17.14 is derived following the steps from (8.59) to (8.64). The equation corresponding to (8.59) is given by

$$\int\int \phi_m \frac{\partial^2 \psi}{\partial y^2} dx\ dy = \int [\phi_m \frac{\partial \psi}{\partial y}]_B^T dx - \int\int \frac{\partial \phi_m}{\partial y} . \frac{\partial \psi}{\partial y} dx\ dy \ . \qquad (1)$$

Further, ϕ_m is now non-zero only on the surface FE, thus,

$$\int\int \phi_m \frac{\partial^2 \psi}{\partial y^2} dx\ dy = \int [\phi_m\ u]_{FE} dx - \int\int \frac{\partial \phi_m}{\partial y} . \frac{\partial \psi}{\partial y} dx\ dy \ . \qquad (2)$$

In (2), the relation $u = \partial \psi / \partial y$ has been used.

Following the derivation of (8.62), one obtains the equivalent of (8.64) which is (17.127) with $g(u) = u_{FE}/\Delta y$ coming from the evaluation of the line integral on the right-hand side of (2).

17.16 The Poisson equation for the Bernoulli variable H can be derived as follows. Using an equivalent procedure to that in Problem 11.2, the x and y momentum equations become

$$\frac{\partial H}{\partial x} = 2\left(-\frac{\partial u}{\partial t} - v\zeta + \frac{1}{Re}\frac{\partial \zeta}{\partial y}\right)$$

$$\text{and} \quad \frac{\partial H}{\partial y} = 2\left(-\frac{\partial v}{\partial t} + u\zeta - \frac{1}{Re}\frac{\partial \zeta}{\partial x}\right),$$

so that, with the introduction of (17.1),

$$\frac{\partial^2 H}{\partial x^2} + \frac{\partial^2 H}{\partial y^2} = 2\left(\frac{\partial (u\zeta)}{\partial y} - \frac{\partial (v\zeta)}{\partial x}\right),$$

as required.

17.17 An approximate factorisation algorithm based on (17.141) is developed as follows. Introducing a three-level fully implicit time discretisation in (17.141), we have,

$$\frac{[(1+\gamma)\Delta\zeta_x^{n+1} - \gamma\Delta\zeta_x^n]}{\Delta t} + \beta(RHS)^{n+1} + (1-\beta)RHS^n = 0 \ , \qquad (1)$$

where

$$\begin{aligned} RHS = {} & u\frac{\partial}{\partial x}(\zeta_x) + \frac{\partial}{\partial y}(v\zeta_x) + \frac{\partial}{\partial z}(w\zeta_x) - \zeta_y\frac{\partial u}{\partial y} - \zeta_z\frac{\partial u}{\partial z} \\ & - \frac{1}{Re}(\frac{\partial^2 \zeta_x}{\partial x^2} + \frac{\partial^2 \zeta_x}{\partial y^2} + \frac{\partial^2 \zeta_x}{\partial z^2}) \ . \end{aligned} \qquad (2)$$

The above equation is obtained from (17.141) by combining the two x derivatives in u and ζ_x. Linearising the term $(RHS)^{n+1}$ as $(RHS)^n + (\partial RHS/\partial\zeta_x)\Delta\zeta_x$, and following the procedure adopted in deriving (17.119), gives

$$\begin{aligned} (1+\gamma)\Big[1 - \Delta t\frac{\beta}{1+\gamma}\Big\{ & u\frac{\partial}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \\ & - \frac{1}{Re}(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2})\Big\}\Big]\Delta\zeta_x^{n+1} = \Delta t(RHS)^n + \gamma\Delta\zeta_x^n \ . \end{aligned} \qquad (3)$$

Now carrying out an approximate factorisation of (3), leads to

$$\left[1 - \frac{\beta\Delta t}{1+\gamma}(uL_x - \frac{1}{Re}L_{xx})\right]\Delta\zeta_x^* = \frac{\Delta t}{1+\gamma}(RHS)^n + \gamma\frac{\Delta\zeta_x^n}{1+\gamma} \ , \qquad (4)$$

$$\left[1 - \frac{\beta\Delta t}{1+\gamma}(L_y v - \frac{1}{Re}L_{yy})\right]\Delta\zeta_x^{**} = \Delta\zeta_x^* \ , \qquad (5)$$

$$\left[1 - \frac{\beta\Delta t}{1+\gamma}(L_z w - \frac{1}{Re}L_{zz})\right]\Delta\zeta_x^{n+1} = \Delta\zeta_x^{**} \ . \qquad (6)$$

We see that L_x operates on $\Delta\zeta_x^*$ on the LHS of (4) and not on $(u\Delta\zeta_x^*)$.

17.18 We shall derive here only the first of the (17.151). The others are obtained in a similar manner.

Consider

$$\zeta_y = \frac{\partial u}{\partial z} - \frac{\partial w}{\partial x}, \quad \zeta_z = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}$$

so that

$$\frac{\partial^2 u}{\partial z^2} + \frac{\partial^2 u}{\partial y^2} = \frac{\partial \zeta_y}{\partial z} - \frac{\partial \zeta_z}{\partial y} + \frac{\partial^2 w}{\partial x \partial z} + \frac{\partial^2 v}{\partial x \partial y} \ .$$

From the continuity equation this becomes

$$\frac{\partial^2 u}{\partial z^2} + \frac{\partial^2 u}{\partial y^2} = \frac{\partial \zeta_y}{\partial z} - \frac{\partial \zeta_z}{\partial y} - \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}\right).$$

Hence,

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = \frac{\partial \zeta_y}{\partial z} - \frac{\partial \zeta_z}{\partial y} \ ,$$

as required.

CTFD Solutions Manual: Chapter 18

Compressible Viscous Flow

18.1 a) Conventional Reynolds averaging, for the continuity equation, is carried out as follows.

Let $\rho = \bar\rho + \rho'$, $u = \bar u + u'$, $v = \bar v + v'$. (1)

Substituting these expressions into the continuity equation we have

$$\frac{\partial\bar\rho}{\partial t} + \frac{\partial\rho'}{\partial t} + \frac{\partial}{\partial x}\left[\bar\rho\bar u + \bar\rho u' + \bar u\rho' + \rho' u'\right] + \frac{\partial}{\partial y}\left[\bar\rho\bar v + \bar\rho v' + \bar v\rho' + \rho' v'\right] = 0 \; . \qquad (2)$$

Now taking the time average of the terms and noting that, $\bar{\rho'} = 0$ etc., we get

$$\frac{\partial\bar\rho}{\partial t} + \frac{\partial}{\partial x}(\bar\rho\bar u + \overline{\rho' u'}) + \frac{\partial}{\partial y}(\bar\rho\bar v + \overline{\rho' v'}) = 0 \; , \qquad (3)$$

as required.

b) Mass-weighted Reynolds averaging, for the continuity equation, is carried out as follows (see 18.1):

Let $u = \tilde u + u''$, $v = \tilde v + v''$, $\rho = \bar\rho + \rho'$ and $\tilde u = \overline{\rho u}/\bar\rho$ etc. (4)

As a result the continuity equation becomes,

$$\frac{\partial\bar\rho}{\partial t} + \frac{\partial\rho'}{\partial t} + \frac{\partial}{\partial x}\left\{\bar\rho\tilde u + \bar\rho u'' + \rho'\tilde u + \rho' u''\right\} + \frac{\partial}{\partial y}\left\{\bar\rho\tilde v + \bar\rho v'' + \rho'\tilde v + \rho' v''\right\} = 0 \; . \qquad (5)$$

Now $\bar\rho\tilde u + \rho'\tilde u + \rho' u'' + \bar\rho u'' = \bar\rho\tilde u + \rho'\tilde u + \rho u''$.

Carrying out the time averaging the above expression reduces to

$\bar\rho\tilde u + \overline{\rho u''}$. Further, $\overline{\rho u''} = 0$, (see 18.1)

Hence, $\bar\rho\tilde u + \rho'\tilde u + \rho' u'' + \bar\rho u'' = \bar\rho\tilde u$ and
$\bar\rho\tilde v + \rho'\tilde v + \rho' v'' + \bar\rho v'' = \bar\rho\tilde v$. (6)

Substituting (3) in (2) and noting that $\bar{\rho'} = 0$, we have

$$\frac{\partial\bar\rho}{\partial t} + \frac{\partial}{\partial x}(\bar\rho\tilde u) + \frac{\partial}{\partial y}(\bar\rho\tilde v) = 0 \; ,$$

as required.

18.2 The x-momentum equation for a two-dimensional, incompressible, viscous flow is given by

$$\rho\{u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}\} = -\frac{\partial p}{\partial x} + \frac{\partial\tau}{\partial y} \; , \qquad (1)$$

where τ is the shear stress.

For a 'Couette model', $\partial u/\partial x = 0$ and v=0. Close to a solid surface, (1) thus reduces to,

$$\frac{\partial\tau}{\partial y} = \frac{\partial p}{\partial x} \; . \qquad (2)$$

Integrating (2) and imposing the boundary condition that at $y = 0$, $\tau = \tau_w$, we have

$$\tau - \tau_w = \frac{\partial p}{\partial x}y \; , \qquad (3)$$

as required.

Derivation of (18.178):

For a turbulent flow, $\tau = \rho(\nu + \nu_\tau)(\partial u/\partial y)$, so that (3) becomes,

$$\rho(\nu + \nu_\tau)\frac{\partial u}{\partial y} - \tau_w = \frac{\partial p}{\partial x}y \; . \qquad (4)$$

Since $\nu_\tau >> \nu$, we have

$$\rho\nu_\tau\frac{\partial u}{\partial y} - \tau_w = \frac{\partial p}{\partial x}y \; . \qquad (5)$$

Introducing the mixing length representation, $\nu_\tau = (\kappa y)^2|(\partial u/\partial y)|$,

$$\rho(\kappa y)^2|\frac{\partial u}{\partial y}|\frac{\partial u}{\partial y} - \tau_w = \frac{\partial p}{\partial x}y \; ,$$

$$\text{so that} \quad \kappa y\frac{\partial u}{\partial y} = \left(\tau_w + \frac{\partial p}{\partial x}y\right)^{1/2}/\rho^{1/2} \qquad (6)$$

as required.

Derivation of (18.23):

If $\partial p/\partial x = 0$, (6) becomes

$$\kappa y \frac{\partial u}{\partial y} = \sqrt{\tau_w/\rho} = u_\tau , \tag{7}$$

where u_τ is the friction velocity $= \sqrt{\tau_w/\rho}$.

Equation (7) can be written as

$$\kappa y^+ \frac{\partial u}{\partial y^+} = u_\tau ,$$

where $y^+ = yu_\tau/\nu$, so that

$$\frac{du}{u_\tau} = \frac{1}{\kappa y^+} dy^+ \quad \text{or} \quad du^+ = \frac{1}{\kappa}\frac{dy^+}{y^+} . \tag{8}$$

Integrating (8), we have

$$u^+ = \frac{1}{\kappa} \ln\, y^+ + C , \tag{9}$$

which is (18.23).

Discretised form of (18.178):

Assume that the computational grid close to the wall is as shown in the figure.

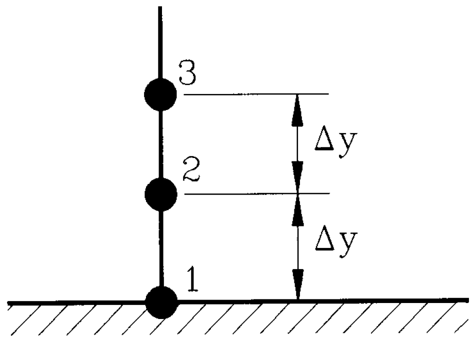

Fig. 1. Grid points close to the wall.

For this grid, (18.178) may be discretised as follows.

$$\kappa y_2 \left[\frac{u_3 - u_1}{2\Delta y}\right] = \left[\mu_1 \left(\frac{u_2 - u_1}{\Delta y}\right) + \left(\frac{\partial p}{\partial x}\right) y_2\right]^{1/2} \Big/ \rho^{1/2} . \tag{10}$$

The velocity at a point close to the wall i.e., 2, can be computed from,

$$\mu_1 \frac{u_2}{\Delta y} = \rho(0.5\kappa u_3)^2 - (\frac{\partial p}{\partial x}) y_2 , \tag{11}$$

where u_1=0 has been substituted. However a consideration of the case $\partial p/\partial x = 0$ implies that (11) will not be very accurate.

18.3 Derivation of (18.29):

Following the order of magnitude analysis carried out in Sect. 16.1.1, we have $u \approx O(1)$, $v \approx O(\delta/L)$, $\partial u/\partial x \approx O(1)$, $\partial u/\partial y \approx O(1/\delta)$, $\partial v/\partial x \approx O(\delta/L)$, $\partial v/\partial y \approx O(1)$. Accordingly,

$$\begin{aligned} RHS = &\frac{\partial}{\partial x}[O(1)] + \frac{\partial}{\partial y}[O(\delta/L)] + \frac{\partial}{\partial x}[O(1)] + \frac{\partial}{\partial y}[O(\delta/L)] \\ &+ \frac{\partial}{\partial x}[O((\delta/L)^2)] + \frac{\partial}{\partial y}\left\{ [\mu(1 - \frac{1}{Pr}) + \mu_\tau(1 - \frac{1}{Pr_T})] u \frac{\partial u}{\partial y} \right\}. \end{aligned} \tag{1}$$

Hence retaining all terms O(L/δ) or bigger, we have

$$RHS = \frac{\partial}{\partial y}\left\{ \left[\mu(1 - \frac{1}{Pr}) + \mu_\tau(1 - \frac{1}{Pr_T})\right] u \frac{\partial u}{\partial y} \right\}.$$

If P_r and P_{r_T} are each assumed equal to 1, RHS = 0, and (18.26) reduces to (18.29).

18.4 The thin layer equations are derived as follows. In (18.7), the viscous terms and heat transfer terms in F are neglected, so that

$$F = \begin{bmatrix} \rho u \\ p + \rho u^2 \\ \rho uv \\ (E + p)u \end{bmatrix}$$

as in (18.32).

The array G has, as its viscous contribution, terms involving τ_{xy} and τ_{yy} which are given by (18.9). Now neglecting the x-derivatives in (18.9), we have,

$$\tau_{xy} = (\mu + \mu_T)\frac{\partial u}{\partial y}, \qquad \tau_{yy} = \frac{4}{3}(\mu + \mu_T)\frac{\partial v}{\partial y} - \frac{2}{3}\rho k^{te} ,$$

thus giving (18.32) i.e. the thin layer equations.

18.5 The truncation error analysis of the explicit MacCormack scheme (18.35 and 18.36) is carried out as follows. The terms, $q^{n+1}, q^*, F^*, G^*, S^*$ are expressed as a Taylor series about q^n, q^n, F^n, G^n, S^n respectively. On substituting these expressions into (18.36) one gets,

$$\begin{aligned} q + q_t\Delta t + q_{tt}\frac{\Delta t^2}{2} + q_{t3}\frac{\Delta t^3}{6} &\\ = q - \frac{\Delta t}{2}\Big(F + F + F_t\Delta t + F_{tt}\frac{\Delta t^2}{2} + F_{t3}\frac{\Delta t^3}{6} + ...\Big)_x &\\ - \frac{\Delta t}{2}\Big(G + G + G_t\Delta t + G_{tt}\frac{\Delta t^2}{2} + G_{t3}\frac{\Delta t^3}{6} + ...\Big)_y &\\ + \frac{\Delta t\Delta x}{2}\Big(-F + F + F_t\Delta t + F_{tt}\frac{\Delta t^2}{2} + F_{t3}\frac{\Delta t^3}{6} + ...\Big)_{xx} &\qquad (1)\\ + \frac{\Delta t\Delta y}{2}\Big(-G + G + G_t\Delta t + G_{tt}\frac{\Delta t^2}{2} + G_{t3}\frac{\Delta t^3}{6} + ...\Big)_{yy} &\\ + \frac{\Delta t}{2}\Big(S + S + S_t\Delta t + S_{tt}\frac{\Delta t^2}{2} + S_{t3}\frac{\Delta t^3}{6} + ...\Big). & \end{aligned}$$

With simplification,

$$\begin{aligned} q_t + F_x + G_y - S = -\frac{\Delta t^2}{2}(F_{ttx} + G_{tty}) &\\ + \Delta t(F_t\Delta x + G_t\Delta y) - S_{tt}\frac{\Delta t^2}{2} + O(H)\ , &\qquad (2) \end{aligned}$$

where the relation $q_{tt} = -F_{xt} - G_{yt} + S_t$ has been made use of. Equation (2) clearly shows that the leading term in the truncation error is of $O(\Delta t^2, \Delta t\Delta x, \Delta t\Delta y)$.

18.6 The explicit MacCormack scheme applied to (18.49) is given by (18.50) and (18.51). It is easily seen that the amplification factor G^* for (18.50) is given by

$$G^* = 1 - a\frac{\Delta t}{\Delta x}(\cos\theta + i\ \sin\theta - 1) + 2\frac{\mu\Delta t}{\Delta x^2}(\cos\theta - 1)\ . \qquad (1)$$

Now expanding terms in (18.51), we have

$$q_j^{n+1} = 0.5(q_j^n + q_j^{*,e} - a\Delta t L_x^- q_j^{*,e} + \mu\Delta t L_{xx} q_j^{*,e}), \qquad (2)$$

so that the amplification factor G for the overall scheme is given by

$$\begin{aligned} G = 0.5 + 0.5\Big[1 - i\sin\theta(\frac{a\Delta t}{\Delta x}) + (\cos\theta - 1)\{\frac{\Delta t}{\Delta x^2}(2\mu - a\Delta x)\}\Big] &\qquad (3)\\ \Big[1 - i\sin\theta(\frac{a\Delta t}{\Delta x}) + (\cos\theta - 1)\{\frac{\Delta t}{\Delta x^2}(2\mu + a\Delta x)\}\Big] & \end{aligned}$$

and required stability condition is

$$\frac{\Delta t}{\Delta x^2}(2\mu + a\Delta x) \le 1\ .$$

18.7 When the scheme (18.43) is written in a predictor corrector format for (18.6), the following formulation is obtained,

$$q'_{j,k} = q^*_{j,k} - \frac{\Delta t_y}{\Delta y}(G^*_{j,k+1} - G^*_{j,k})$$

$$\text{and}\quad q^{**}_{j,k} = 0.5(q^*_{j,k} + q'_{j,k}) - 0.5\frac{\Delta t_y}{\Delta y}(G'_{j,k} - G'_{j,k-1}).$$

18.8 For the Wambecq scheme (18.47), with c=1 and b=− 0.5, we get

$$\begin{aligned} \Delta u^1 &= \Delta t\mu\ L_{xx}\ u^n\ , \\ \Delta u^2 &= \Delta t\mu\ L_{xx}(u^n + \Delta u^1)\ , \qquad (1)\\ \Delta u^3 &= \Delta t\mu\ L_{xx}(1.5u^n - 0.5u^*)\ . \end{aligned}$$

But $\Delta u^{n+1} = (2\ \Delta u^1(\Delta u^1, \Delta u^3) - \Delta u^3(\Delta u^1, \Delta u^1))/(\Delta u^3, \Delta u^3)$.

Given that $(\Delta u^1,\ \Delta u^3) = n\Delta u^1\Delta u^3$ etc, so that the above equation becomes, on simplification,

$$\Delta u^{n+1} = \frac{(\Delta u^1)^2}{\Delta u^3} = \frac{\Delta t\mu(L_{xx}\ u^n)^2}{L_{xx}(1.5u^n - 0.5u^*)}\ , \qquad (2)$$

as required.

Implementation of the Wambecq scheme:

Equation (18.47) on which the Wambecq scheme is implemented, is identical to the diffusion equation studied in Chapter 7. Program DIFEX is easily modified to do this. See the following listing of the program which implements the true Wambecq scheme and the one given by (18.180). On running the program the following solution errors are computed, at $t = 9$ secs. In the table, TWS denotes True Wambecq Scheme (with b=−0.5, c=1), PRESENT denotes the scheme given by (18.180).

Δx	TWS	PRESENT
0.2	0.03472	0.0644
0.1	0.02069	0.00907
0.05	0.007543	0.00384

To check the stability of the two schemes, the programs were run for times up to $t = 90$ sec. for various values of $s(\alpha\Delta t/\Delta x^2$ or $\mu\Delta t/\Delta x^2)$. The computed solution errors are given below.

s	TWS	PRESENT
0.5	0.00093	0.001128
1.0	0.1627	0.9188
2.0	1.194	3.462

Thus, we find that though the schemes are stable for values of $s = 1.0$, 2.0 etc. the solution error increases sharply. For best results smaller values of $s(\leq 0.5)$ should be chosen. This type of restriction is similar to that for the DuFort-Frankel scheme (Sect. 7.1.2). The following code replaces lines 1 to 112 of program DIFEX.

Modifications to DIFEX for Problem 18.8

```
C     PR18P8.
C     THIS PROGRAM SOLVES THE DIFFUSION 1D TRANSIENT
C     HEAT CONDUCTION EQUATION USING WAMBECQ SCHEME.
C     NWAM=1, TRUE WAMBECQ SCHEME (18.47) USED.
C     NWAM=0, MODIFIED WAMBECQ SCHEME (18.180) USED.
C
      DIMENSION TN(41),DUM(41),TD(41),X(41),TE(41),TOL(41),
     1 DT1(41),DT2(41),DT3(41)
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      OPEN(1,FILE='PR188.DAT')
      OPEN(6,FILE='PR188.OUT')
      READ(1,*)JMAX,MAXEX,NMAX,ALPH,S,TMAX
      READ(1,*)B,C,NWAM
C
      PI = 3.1415927
      JMAP = JMAX - 1
      AJM = JMAP
      DELX = 1./AJM
      DELT = DELX*DELX*S/ALPH
C
      IF(NWAM .EQ. 1) THEN
      WRITE(6,*)'APPLICATION OF TRUE WAMBECQ SCHEME
     1  TO DIFFUSION EQUATION'
      ELSE
      WRITE(6,*)'APPLICATION OF MODIFIED WAMBECQ SCHEME
     1  TO DIFFUSION EQUATION'
      ENDIF
      WRITE(6,5)JMAX,MAXEX,NMAX,TMAX
    5 FORMAT(' JMAX=',I5,'  MAXEX=',I5,'  NMAX=',I5,
     1'  TMAX=',F5.2)
      WRITE(6,6)S,ALPH,DELT,DELX,B,C
    6 FORMAT(' S=',F5.3,'  ALPH = ',E10.3,'  DELT = ',E10.3,
     1'  DELX = ',E10.3,' B= ',F5.2,' C= ',F5.3,//)
C
      DO 8 J = 2,JMAP
      TN(J) = 0.0
```

```
    8 CONTINUE
    9 CONTINUE
C
C     SET BOUNDARY CONDITIONS
C
      TOL(1) = 1.
      TOL(JMAX) = 1.
      TN(1) = 1.
      TN(JMAX) = 1.
      DUM(1) = 1.
      DUM(JMAX) = 1.
      TD(1) = 100.*TN(1)
      TD(JMAX) = 100.*TN(JMAX)
      N = 0
C
C     EACH TIME STEP STARTS AT STATEMENT 10
C
   10 N = N + 1
C
      DO 200 J=2,JMAP
  200 DT1(J)=S*(TN(J-1)-2.*TN(J)+TN(J+1))
      IF(NWAM .EQ. 1) THEN
C     TRUE WAMBECQ SCHEME.
      DO 201 J=2,JMAP
  201 DUM(J)=TN(J)+C*DT1(J)
C
      DO 202 J=2,JMAP
  202 DT2(J)=S*(DUM(J-1)-2.*DUM(J)+DUM(J+1))
      DO 203 J=2,JMAP
  203 DT3(J)=(1.-B)*DT1(J)+B*DT2(J)
C     WEIGHTING FACTORS.
C
      WHT13=0.0
      WHT11=0.0
      WHT33=0.0
      DO 204 J=2,JMAP
      WHT13=WHT13 + DT1(J)*DT3(J)
      WHT11=WHT11 + DT1(J)*DT1(J)
  204 WHT33=WHT33 + DT3(J)*DT3(J)
C
      DO 210 J=2,JMAP
  210 TN(J)=TN(J)+(2.*DT1(J)*WHT13 - DT3(J)*WHT11)/WHT33
C     MODIFED WAMBECQ SCHEME.
      ELSE
      DO 220 J=2,JMAP
      DUM(J)=TN(J)+DT1(J)
  220 DUM(J)=1.5*TN(J)-0.5*DUM(J)
C
      DO 221 J=2,JMAP
  221 DT2(J)=(DUM(J-1)-2.*DUM(J)+DUM(J+1))/(DELX*DELX)
C
      DO 222 J=2,JMAP
      IF(DT2(J).NE.0.0)
     1TN(J)=TN(J)+DT1(J)*DT1(J)/DT2(J)/DELT/ALPH
  222 CONTINUE
      ENDIF
      DO 19 J = 2,JMAP
   19 TD(J) = 100.*TN(J)
      T = T + DELT
C
C     IF MAXIMUM TIME OR MAXIMUM NUMBER OF TIME-STEPS
C     EXCEEDED EXIT FROM LOOP
C
      IF(N .GE. NMAX)GOTO 21
      IF(T .LT. TMAX)GOTO 10
C
C     OBTAIN EXACT SOLUTION AND COMPARE
```

18.9 Equations (18.53) and (18.54) can be rewritten as

$$\Delta q_j^{*,i} = \Delta q_j^{*,e} + \lambda\frac{\Delta t}{\Delta x}(\Delta q_{j+1}^{*,i} - \Delta q_j^{*,i}), \tag{1}$$

$$\Delta q_j^{n+1,i} = \Delta q_j^{n+1,e} + \lambda\frac{\Delta t}{\Delta x}(\Delta q_{j-1}^{n+1,i} - \Delta q_j^{n+1,i}). \tag{2}$$

Expanding the implicit terms on the RHS as a Taylor series, we have

$$\begin{aligned}\Delta q_j^{*,i} &= \Delta q_j^{*,e} + \lambda\frac{\Delta t}{\Delta x}\{\Delta q^{*,i} + \frac{\partial \Delta q^{*,i}}{\partial x}\Delta x + \frac{\partial^2 \Delta q^{*,i}}{\partial x^2}\frac{\Delta x^2}{2} + \dots - \Delta q^{*,i}\}\\ &= \Delta q_j^{*,e} + \lambda\ \Delta t^2\frac{\partial^2 q}{\partial x\partial t} + \frac{\lambda\cdot\Delta t^2\Delta x}{2}\frac{\partial^3 q}{\partial x^2\partial t} + \dots\end{aligned} \tag{3}$$

Clearly, the term $0.5\lambda\Delta t^2\ \Delta x(\partial^3 q/\partial x^2\partial t)$ is of third order in time, since Δx can always be expressed in terms of Δt through a relation like $\Delta x = a\Delta t/C$.

Similarly,

$$\Delta q_j^{n+1,i} = \Delta q_j^{*,e} - \lambda\Delta t^2\frac{\partial^2 q}{\partial x\partial t} + \frac{\lambda\Delta t^2\Delta x}{2}\frac{\partial^3 q}{\partial x^2\partial t} + \dots \tag{4}$$

$$\begin{aligned}\text{Now}\quad q_j^{n+1} &= 0.5(q_j^n + q_j^{*,i} + \Delta q_j^{n+1,i})\\ &= 0.5(q_j^n + q_j^n + \Delta q_j^{*,i} + \Delta q_j^{n+1,i})\\ &= 0.5(q_j^n + q_j^* + \Delta q_j^{*,e}) + O(\Delta t^3)\ .\end{aligned} \tag{5}$$

This shows that the implicit terms act as third order perturbations to the explicit ones.

18.10 The amplification factor $G^{*,i}$ for (18.53) is given by

$$\begin{aligned}&(1+\lambda\frac{\Delta t}{\Delta x})(G^{*,i}-1) = -a\frac{\Delta t}{\Delta x}(\cos\theta - 1 + i\sin\theta)\\ &+2\mu\frac{\Delta t}{\Delta x^2}(\cos\theta-1) + \lambda\frac{\Delta t}{\Delta x}(\cos\theta + i\sin\theta)(G^{*,i}-1)\ .\end{aligned} \tag{1}$$

i.e.

$$\begin{aligned}G^{*,i} = &\left\{\frac{\Delta x}{\Delta t} + (\cos\theta-1)(\frac{2\mu}{\Delta x} - a - \lambda) - i\sin\theta(\lambda+a)\right\}\\ &\Big/\left\{\frac{\Delta x}{\Delta t} - \lambda(\cos\theta-1) - i\lambda\sin\theta\right\}.\end{aligned} \tag{2}$$

Similarly for (18.54) we have,

$$\begin{aligned}&(1+\lambda\frac{\Delta t}{\Delta x})(\frac{2G-G^{*,i}-1}{2}) = -a\frac{\Delta t}{\Delta x}(1-\cos\theta+i\sin\theta)G\\ &+2\mu\frac{\Delta t}{\Delta x^2}(\cos\theta-1)G + \lambda\frac{\Delta t}{\Delta x}(\cos\theta - i\sin\theta)(\frac{2G-G^{*,i}-1}{2}).\end{aligned} \tag{3}$$

Simplifying (3), we find

$$G = 0.5\left\{1 + (\frac{A}{B})\cdot(\frac{C}{D})\right\}, \tag{4}$$

where,

$$\begin{aligned}A &= \frac{\Delta x}{\Delta t} + (\cos\theta-1)(\frac{2\mu}{\Delta x} - a - \lambda) - i(\lambda+a)\sin\theta\ ,\\ B &= \frac{\Delta x}{\Delta t} - \lambda(\cos\theta-1) - i\lambda\sin\theta\ ,\\ C &= 0.5\frac{\Delta x}{\Delta t} + (\cos\theta-1)(\frac{2\mu}{\Delta x} - a - \frac{\lambda}{2}) - i(\frac{\lambda}{2}+a)\sin\theta\ ,\\ D &= \frac{\Delta x}{\Delta t} - \lambda(\cos\theta-1) + i\lambda\sin\theta\ .\end{aligned}$$

Given the complexity of the expression for G it is preferable to either use a symbolic algebra computer program or to evaluate G numerically. This can be carried out by assigning a range of values to a, $2\mu/\Delta x$ and $\Delta x/\Delta t$. For each combination a range of values of λ is chosen such that $\lambda \geq a + 2\mu/\Delta x - \Delta x/\Delta t$. For every combination θ is chosen such that $0 \leq \theta \leq 2\pi$ and $|G|$ is evaluated. If $|G| \leq 1.0$ then (18.55) is confirmed.

18.11 The predictor stage of the implicit MacCormack scheme for the Navier-Stokes equations is given by (18.58) which is implemented as

$$(\underline{\mathbf{I}} - \Delta t\ L_x^+ \underline{\mathbf{A}}^m)\Delta \mathbf{q}_{j,k}^{*,i} = \Delta \mathbf{q}_{j,k}^{*,e} \tag{1}$$

$$\text{and}\quad (\underline{\mathbf{I}} - \Delta t\ L_y^+ \underline{\mathbf{B}}^m)\Delta \mathbf{q}_{j,k}^{**,i} = \Delta \mathbf{q}_{j,k}^{*,i}\ . \tag{2}$$

When expanded, (1) is given by

$$\Delta \mathbf{q}_{j,k}^{*,i} - \frac{\Delta t}{\Delta x}\underline{\mathbf{A}}_{j+1,k}^m \Delta \mathbf{q}_{j+1,k}^{*,i} + \frac{\Delta t}{\Delta x}\underline{\mathbf{A}}_{j,k}^m \Delta \mathbf{q}_{j,k}^{*,i} = \Delta \mathbf{q}_{j,k}^{*,e}\ . \tag{3}$$

which reduces to (18.65) in the text.

Similarly, (2) can be implemented as

$$(\underline{\mathbf{I}} + \frac{\Delta t}{\Delta y}\underline{\mathbf{B}}_{j,k}^m)\Delta \mathbf{q}_{j,k}^{**,i} = \Delta \mathbf{q}_{j,k}^{*,i} + \frac{\Delta t}{\Delta y}\underline{\mathbf{B}}_{j,k+1}^m \Delta \mathbf{q}_{j,k+1}^{**,i}\ , \tag{4}$$

which is (18.67) in the text.

Let the RHS of (4) = $\mathbf{W}$, so that

$$(\underline{\mathbf{I}} + \frac{\Delta t}{\Delta y}\underline{\mathbf{B}}^m)\Delta \mathbf{q}_{j,k}^{**,i} = \mathbf{W}\ . \tag{5}$$

From (18.62), $\underline{\mathbf{B}} = (\underline{\mathbf{T}}^B)^{-1}\underline{\mathbf{D}}^B\underline{\mathbf{T}}^B$,

so that (5) becomes

$$(\underline{\mathbf{I}} + \tfrac{\Delta t}{\Delta y}(\underline{\mathbf{T}}^B)^{-1}\underline{\mathbf{D}}^B\underline{\mathbf{T}}^B)\Delta \mathbf{q}_{j,k}^{**,i} = \mathbf{W}\ .$$

Multiplying both sides by $\underline{\mathbf{T}}^B$, and simplifying,

$$\underline{\mathbf{T}}^B \Delta \mathbf{q}_{j,k}^{**,i}(1 + \frac{\Delta t}{\Delta y}\underline{\mathbf{D}}^B) = \mathbf{W}\underline{\mathbf{T}}^B,$$

$$\underline{\mathbf{T}}^B \Delta \mathbf{q}_{j,k}^{**,i} = \left(1 + \frac{\Delta t}{\Delta y}\underline{\mathbf{D}}^B\right)^{-1}\mathbf{W}\underline{\mathbf{T}}^B = \mathbf{Y}\ .$$

$$\Delta \mathbf{q}_{j,k}^{**,i} = (\underline{\mathbf{T}}^B)^{-1}\mathbf{Y}\ ,$$

as required.

18.12 The transport equation (9.81) is given by

$$\frac{\partial T}{\partial t} + u\frac{\partial T}{\partial x} + v\frac{\partial T}{\partial y} - \alpha_x\frac{\partial^2 T}{\partial x^2} - \alpha_y\frac{\partial^2 y}{\partial y^2} = 0\ . \tag{1}$$

The Beam and Warming algorithm is applied to this equation in the following manner. Writing (1) as

$$\frac{\partial T}{\partial t} = RHS\ . \tag{2}$$

Applying the generalised three-level scheme (Sect. 8.2.3),

$$(1+\alpha)\Delta T^{n+1} - \alpha\Delta T^n = \Delta t\{\beta \cdot RHS^{n+1} + (1-\beta)RHS^n\}. \tag{3}$$

Now, linearising RHS^{n+1}, which is a function of T_x, T_y, T_{xx} and T_{yy}, we have

$$\begin{aligned} RHS^{n+1} = RHS^n &- u\frac{\partial}{\partial x}(\Delta T)^{n+1} - v\frac{\partial}{\partial y}(\Delta T)^{n+1} \\ &+ \alpha_x\frac{\partial^2}{\partial x^2}(\Delta T)^{n+1} + \alpha_y\frac{\partial^2}{\partial y^2}(\Delta T)^{n+1}. \end{aligned} \tag{4}$$

Substituting (4) in (3), and simplifying we have

$$\begin{aligned} &\left\{1 + \frac{\beta\Delta t}{1+\alpha}\left(u\frac{\partial}{\partial x} + v\frac{\partial}{\partial y} - \alpha_x\frac{\partial^2}{\partial x^2} - \alpha_y\frac{\partial^2}{\partial y^2}\right)\right\}\Delta T^{n+1} \\ &= \frac{\Delta t}{1+\alpha}RHS^n + \frac{\alpha}{1+\alpha}\Delta T^n. \end{aligned} \tag{5}$$

Carrying out the approximate factorisation,

$$\left\{1 + \frac{\beta\Delta t}{1+\alpha}\left(u\frac{\partial}{\partial x} - \alpha_x\frac{\partial^2}{\partial x^2}\right)\right\}\Delta T^* = \frac{\Delta t}{1+\alpha}RHS^n + \frac{\alpha}{1+\alpha}\Delta T^n \tag{6}$$

and

$$\left\{1 + \frac{\beta\Delta t}{1+\alpha}\left(v\frac{\partial}{\partial y} - \alpha_y\frac{\partial^2}{\partial y^2}\right)\right\}\Delta T^{n+1} = \Delta T^n. \tag{7}$$

It may be seen that (6) and (7) are equivalent to (18.78) and (18.79) with $q = T$, $A = u$, $P = 0$, $R_x = 0$, $R = \alpha_x$, etc.

Starting from (9.88) and (9.89) the above equations are obtained when $M_x = (0,1,0)$ and $M_y^T = (0,1,0)$ and $\gamma = \alpha$.

18.13 The approximate factorisation algorithm for (18.84) can be derived as follows.

The given equation in semi-discrete form is

$$\begin{aligned} M_x \otimes M_y\frac{\partial \mathbf{q}}{\partial t} &+ M_y \otimes L_x\mathbf{F}^I + M_x \otimes L_y\mathbf{G}^I \\ &= M_y \otimes L_{xx}\mathbf{R} + L_x \otimes L_y\mathbf{S} + M_x \otimes L_{yy}\mathbf{T}\ . \end{aligned} \tag{1}$$

A three-level algorithm to advance the solution in time is introduced as in the previous problem. With the usual linearisation one gets,

$$\begin{aligned} \mathbf{RHS}^{n+1} = \mathbf{RHS}^n &+ \left\{-M_y \otimes L_x\frac{\partial \mathbf{F}^I}{\partial \mathbf{q}} - M_x \otimes L_y\frac{\partial \mathbf{G}^I}{\partial \mathbf{q}} + \right. \\ &\left. M_y \otimes L_{xx}\frac{\partial \mathbf{R}}{\partial \mathbf{q}} + M_x \otimes L_{yy}\frac{\partial \mathbf{T}}{\partial \mathbf{q}}\right\}\Delta \mathbf{q}^{n+1} + \left\{L_x \otimes L_y\frac{\partial \mathbf{S}}{\partial \mathbf{q}}\right\}\Delta \mathbf{q}^n. \end{aligned} \tag{2}$$

As usual the cross-derivative term $\{L_x \otimes L_y(\partial \mathbf{S}/\partial \mathbf{q})\}$ is evaluated explicity i.e. at time level n.

As a consequence the algorithm is ,

$$\begin{aligned}&\Big\{M_x \otimes M_y - \frac{\beta \Delta t}{1+\alpha}\Big(M_y \otimes L_{xx}\frac{\partial \mathbf{R}}{\partial \mathbf{q}} + M_x \otimes L_{yy}\frac{\partial \mathbf{T}}{\partial \mathbf{q}}\\ &\qquad - M_y \otimes L_x\underline{\mathbf{A}} - M_x \otimes L_y \underline{\mathbf{B}}\Big)\Big\}\Delta \mathbf{q}^{n+1} \\ &\qquad = \frac{\Delta t}{1+\alpha}\mathbf{RHS}^n + \frac{\beta \Delta t}{1+\alpha} L_x \otimes L_y \frac{\partial \mathbf{S}}{\partial \mathbf{q}} + \frac{M_x \otimes M_y}{1+\alpha}\alpha\ \Delta \mathbf{q}^n ,\end{aligned} \tag{3}$$

where $\underline{\mathbf{A}} = (\partial \mathbf{F}^I/\partial \mathbf{q})$, $\underline{\mathbf{B}} = (\partial \mathbf{G}^I / \partial \mathbf{q})$.

Carrying out the approximation on (3), (18.85) and (18.86) are obtained.

This may be verified by substituting for $\Delta \mathbf{q}^*$ from (18.86) in (18.85) when we obtain

$$\begin{aligned}&\left[M_x - \frac{\beta}{1+\alpha}\Delta t\left(L_{xx}\frac{\partial \mathbf{R}}{\partial \mathbf{q}} - L_x \underline{\mathbf{A}}\right)\right] \\ &\left[M_y - \frac{\beta}{1+\alpha}\Delta t\left(L_{yy}\frac{\partial \mathbf{T}}{\partial \mathbf{q}} - L_y \underline{\mathbf{B}}\right)\right]\ \Delta \mathbf{q}^{n+1} = \mathbf{RHS} \quad \text{in} \quad (18.85).\end{aligned} \tag{4}$$

Expanding the left-hand side of (4), we have

$$\begin{aligned}&LHS\ of\ (4) = LHS\ of\ (3) + \\ &\quad \left(\frac{\beta \Delta t}{1+\alpha}\right)^2 \left(L_{xx}\frac{\partial \mathbf{R}}{\partial \mathbf{q}} - L_x \underline{\mathbf{A}}\right)\left(L_{yy}\frac{\partial \mathbf{T}}{\partial \mathbf{q}} - L_y \underline{\mathbf{B}}\right)\Delta \mathbf{q}^{n+1} \ .\end{aligned} \tag{5}$$

The extra terms in (3) are a consequence of approximate factorisation.

18.14 The given equation is

$$L_x A \Delta q_j^{n+1} \rightarrow L_x^-(A_j^+ \Delta q_j^{n+1}) + L_x^+(A_j^- \Delta q_j^{n+1}). \tag{1}$$

Expanding each of the terms as a Taylor series, we have

$$\begin{aligned}&(\Delta q)_x + (Aq)_{x^3}\frac{\Delta x^2}{6} + (Aq)_{x^5}\frac{\Delta x^4}{24} \rightarrow \\ &(A^+q + A^-q)_x + (A^-q - A^+q)_{xx}\frac{\Delta x}{2} + (A^+q + A^-q)_{x^3}\frac{\Delta x^2}{6} + \ldots .\end{aligned} \tag{2}$$

Clearly $0.5(A^-q - A^+q)_{xx}\Delta x$ is the additional dissipative term introduced on the RHS. Thus for unsteady problems it is important to ensure that this term is small.

18.15 From (18.110) we have

$$\mathbf{S} = \frac{1}{ReJ}\begin{bmatrix} 0 \\ \eta_x \tau_{xx} + \eta_y \tau_{xy} \\ \eta_x \tau_{xy} + \eta_y \tau_{yy} \\ \eta_x R_4 + \eta_y S_4 \end{bmatrix} . \tag{1}$$

Each term in (1) is considered separately . Using the chain rule and dropping the ξ-derivatives

$$\text{we have}\quad \frac{\partial u}{\partial x} = \eta_x u_\eta \quad \text{and} \quad \frac{\partial v}{\partial y} = \eta_y v_\eta , \tag{2}$$

so that

$$\begin{aligned}\tau_{xx} &= 2\mu(\eta_x u_\eta) - \frac{2}{3}\mu(\eta_x u_\eta + \eta_y v_\eta) \ , \\ \tau_{yy} &= 2\mu(\eta_y v_\eta) - \frac{2}{3}\mu(\eta_x u_\eta + \eta_y v_\eta) \ , \\ \tau_{xy} &= \mu(\eta_y u_\eta + \eta_x v_\eta) \ , \\ R_4 &= u\left[2\mu(\eta_x u_\eta) - \frac{2}{3}\mu(\eta_x u_\eta + \eta_y v_\eta)\right] \\ &+ v\mu(\eta_y u_\eta + \eta_x v_\eta) + \frac{k}{(r-1)P_r}\frac{\partial a^2}{\partial x} \ , \\ S_4 &= u\mu(\eta_y u_\eta + \eta_x v_\eta) + v\left[2\mu(\eta_y v_\eta) - \frac{2}{3}\mu(\eta_x u_\eta + \eta_y v_\eta)\right] \\ &+ \frac{k}{(r-1)P_r}\frac{\partial a^2}{\partial y} \ .\end{aligned} \tag{3}$$

Substituting (3) into (1), (18.116) is obtained.

18.16 To get the components of $\hat{\underline{\mathbf{M}}}$ we have to differentiate the components of $\hat{\mathbf{S}}$ with respect to $\hat{\mathbf{q}}$.

We have $\hat{\mathbf{S}} = \{S_1, S_2, S_3, S_4\}^T$.

A typical element of $\hat{\underline{\mathbf{M}}}$, say $m_{22} = J(\partial S_2/\partial \rho u)$,

$S_2 = (1/Re\ J)[\alpha_1 u_\eta + \alpha_2 v_\eta]$,

where α_1, α_2 are given by (18.121).

$$\begin{aligned}m_{22} &= \frac{J}{ReJ}\frac{\partial}{\partial(\rho u)}[\alpha_1 u_\eta + \alpha_2 v_\eta] \\ &= \frac{1}{Re}\frac{\partial}{\partial(\rho u)}\left[\alpha_1\frac{(\rho u)_\eta - u\rho_\eta}{\rho} + \alpha_2\frac{(\rho v)_\eta - v\rho_\eta}{\rho}\right] \\ &= \frac{1}{\rho Re}\frac{\partial}{\partial(\rho u)}\left[\alpha_1\left((\rho u)_\eta - \frac{\rho u \cdot \rho_\eta}{\rho}\right) + \alpha_2\left((\rho v)_\eta - \frac{\rho v\cdot \rho_\eta}{\rho}\right)\right].\end{aligned} \tag{1}$$

This reduces to $\dfrac{\alpha_1}{Re}\dfrac{\partial}{\partial \eta}\left(\dfrac{1}{\rho}\right)$

as required.

18.17 The governing equation for transonic, viscous flow (18.80) is given in generalised coordinates as (see Sect. 12.3.2):

$$\frac{\partial \hat{\mathbf{q}}}{\partial t}+\frac{\partial \hat{\mathbf{F}}}{\partial \xi}+\frac{\partial \hat{\mathbf{G}}}{\partial \eta}=\frac{\partial^2 \hat{\mathbf{R}}}{\partial \xi^2}+\frac{\partial^2 \hat{\mathbf{S}}}{\partial \xi \partial \eta}+\frac{\partial^2 \hat{\mathbf{T}}}{\partial \eta^2}\ , \tag{1}$$

where $\hat{\mathbf{q}}$, $\hat{\mathbf{F}}$ etc are given by (12.70).

For example,

$$\hat{\mathbf{F}}=\frac{\xi_x \mathbf{F}^I+\xi_y \mathbf{G}^I}{J}+\frac{\xi_{xx}\mathbf{R}+\xi_{xy}\mathbf{S}+\xi_{yy}\mathbf{T}}{J}.$$

Substituting for various terms ($\mathbf{F}, \mathbf{G}, \mathbf{R}$ and $\mathbf{S}$) and simplifying, we get (18.138). For example,

$$\begin{aligned}\hat{F}_2&=\frac{1}{J}\left\{\xi_x F_2^I+\xi_y G_2^I+\xi_{xx}R_2+\xi_{xy}S_2+\xi_{yy}T_2\right\}\\&=\frac{1}{J}\left\{\xi_x(p+\rho u^2)+\xi_y(\rho u v)+\xi_{xx}(4\mu_e u/3)+\xi_{xy}(\mu_e v/3)+\xi_{yy}(\mu_e u)\right\}\\&=\frac{1}{J}\left\{\xi_x p+\rho u(\xi_x u+\xi_y v)+(4\xi_{xx}/3+\xi_{yy})\mu_e u+(\xi_{xy}/3)\mu_e v\right\}\ .\end{aligned}$$

Equation (18.139) is similarly derived.

Considering now the term $\hat{\mathbf{R}}$,

$$\hat{\mathbf{R}}=\frac{\xi_x^2\mathbf{R}+\xi_x\xi_y\mathbf{S}+\xi_y^2\mathbf{T}}{J}.$$

Substituting for the various terms the required expression for $\hat{\mathbf{R}}$ is obtained.

For example,

$$\begin{aligned}\hat{R}_2&=\frac{1}{J}\left\{\xi_x^2R_2+\xi_x\xi_yS_2+\xi_y^2T_2\right\}\\&=\frac{1}{J}\left\{\xi_x^2(4\mu_e u/3)+\xi_x\xi_y(\mu_e v/3)+\xi_y^2(\mu_e u)\right\}\ .\end{aligned}$$

Equations for $\hat{\mathbf{T}}$ and $\hat{\mathbf{S}}$ are derived in the same manner.

18.18 The one-dimensional transport equation with fourth-order dissipative terms is given by

$$\frac{\partial T}{\partial t}+u\frac{\partial T}{\partial x}-\alpha\frac{\partial^2 T}{\partial x^2}+\epsilon_E\Delta x^4\frac{\partial^4 T}{\partial x^4}\ =\ 0\ . \tag{1}$$

Applying the general two-level scheme to (1), we have

$$\frac{\Delta T^{n+1}}{\Delta t}=\beta\ RHS^{n+1}+(1-\beta)RHS^n\ . \tag{2}$$

With centred differences the above algorithm for (1) reduces to,

$$\begin{aligned}&T_j^{n+1}\left[1+2\beta s+6\beta\epsilon_E\Delta x^4\Delta t\right]-\beta\Big[-\frac{C}{2}(T_{j+1}-T_{j-1})+\\&\qquad s(T_{j+1}+T_{j-1})-\epsilon_E\Delta x^4\Delta t(T_{j-2}-4T_{j-1}-4T_{j+1}+T_{j+2})\Big]^{n+1}\\&\qquad=T_j^n\left[1+2(1-\beta)s+6(1-\beta)\epsilon_E\Delta x^4\Delta t\right]+\\&\qquad(1-\beta)\Big[-\frac{C}{2}(T_{j+1}-T_{j-1})+s(T_{j+1}+T_{j-1})-\\&\qquad\epsilon_E\Delta x^4\Delta t(T_{j-2}-4T_{j-1}-4T_{j+1}+T_{j+2})\Big]^n.\end{aligned} \tag{3}$$

Applying the von Neumann method,

$$\begin{aligned}&G\Big\{\left(1+2\beta s+6\beta\epsilon_E\Delta x^4\Delta t\right)-\beta\big(-ic\sin\theta+s\cos\theta\\&\quad-\epsilon_E\Delta x^4\Delta t(2\cos 2\theta-4\cos 4\theta)\big)\Big\}=\\&\Big(1+2(1-\beta)s+6(1-\beta)\epsilon_E\Delta x^4\Delta t\Big)\\&\quad+(1-\beta)\Big(-ic\sin\theta+s\cos\theta-\epsilon_E\Delta x^4\Delta t(2\cos 2\theta-4\cos 4\theta)\Big)\ .\end{aligned}$$

Although an explicit expression for G is obtained it is difficult to complete the von Neumann stability analysis by hand. One can either use symbolic algebra or evaluate G numerically for a range of values of the following parameters: $\beta, \epsilon_E\Delta x^2\Delta t, s$ and θ. For stability it is necessary to ensure that $-1.0\le G\le 1.0$. For specific choices it is possible to complete the analysis by hand. For example with $s=0$ and $\theta=0$ the amplification factor reduces to

$$G=\{1+8E(1-\beta)\}/\{1+4E\beta\},$$

where $E=\epsilon_E\Delta x^2\Delta t$. Requiring $G>-1.0$ leads to $\beta<2+0.5/E$ and requiring $G<1.0$ leads to $\beta>1/12$.

18.19 Equation (18.160) is given by

$$\begin{aligned}\tilde{f}_{j+1/2}&=0.5(f_j+f_{j+1})-0.5[\phi(r)C]_{j+1/2}\Delta f_{j+1/2}\\&\quad-0.5\sigma[1-\phi(r)]_{j+1/2}\Delta f_{j+1/2}\end{aligned} \tag{1}$$

and

$$\begin{aligned}\tilde{f}_{j-1/2}&=0.5(f_{j-1}+f_j)-0.5[\phi(r)C]_{j-1/2}\Delta f_{j-1/2}\\&\quad-0.5\sigma[1-\phi(r)]_{j-1/2}\Delta f_{j-1/2}\ .\end{aligned} \tag{2}$$

Substituting these expressions into (18.159), we have

$$\begin{aligned}\rho^{n+1}=&\rho^n-\frac{\Delta t}{\Delta x}0.5(f_{j+1}-f_{j-1})\\&+\left\{0.5(\phi(r)C)-0.5\sigma(1-\phi(r))\right\}_{j+1/2}\Delta f_{j+1/2}\\&-\left\{0.5(\phi(r)C)-0.5\sigma(1-\phi(r))\right\}_{j-1/2}\Delta f_{j-1/2}\ .\end{aligned} \tag{3}$$

Writing (3) as

$$\rho^{n+1} = \rho^n - 0.5\frac{\Delta t}{\Delta x}(f_{j+1} - f_{j-1}) + T_{j+1/2}\Delta f_{j+1/2} - T_{j-1/2}\Delta f_{j-1/2}$$

and expanding the last two terms as a Taylor series about 'j' we see that,

$$\rho^{n+1} = \rho^n - 0.5\frac{\Delta t}{\Delta x}(f_{j+1} - f_{j-1}) + T_x f_x \Delta t \Delta x + T f_{xx}\frac{\Delta x \Delta t}{2} + O(H).$$

Thus (18.159) along with (18.160) is equivalent to the use of centred differences with a second-order dissipation term.

Substituting (18.162) into (18.161) we have

$$\begin{aligned}\rho^{n+1}+&\\ &\beta\frac{\Delta t}{\Delta x}\Big\{0.5(f_j + f_{j+1}) - 0.5|u_{j+1/2}|[1-\phi(r)]_{j+1/2}\Delta\rho_{j+1/2}\\ &- 0.5(f_{j-1} + f_j) + 0.5|u_{j-1/2}|[1-\phi(r)]_{j-1/2}\Delta\rho_{j-1/2}\Big\}^{n+1}\\ &= \rho^n - (1-\beta)\frac{\Delta t}{\Delta x}\Big\{\qquad\Big\}^n,\end{aligned} \tag{4}$$

i.e.

$$\begin{aligned}\rho^{n+1} + \beta\frac{\Delta t}{\Delta x}\Big\{&0.5(f_{j+1} - f_{j-1}) - 0.5|u_{j+1/2}|[1-\phi(r)]_{j+1/2}\Delta\rho_{j+1/2} +\\ &0.5|u_{j-1/2}|[1-\phi(r)]_{j-1/2}\Delta\rho_{j-1/2}\Big\}^{n+1}\\ &= \rho^n - (1-\beta)\frac{\Delta t}{\Delta x}\Big\{\qquad\Big\}^n.\end{aligned} \tag{5}$$

Now expanding the terms $|u_{j+1/2}|, [1-\phi(r)]_{j+1/2}, \Delta\rho_{j+1/2}, |u_{j-1/2}|,$ $[1-\phi(r)]_{j-1/2}$ and $\Delta\rho_{j-1/2}$ as a Taylor series about 'j' and simplifying it can be shown that (5) is also equivalent to the use of centred differences with a second-order dissipation term.

Computational Techniques for Fluid Dynamics

This manual, which complements C.A.J. Fletcher's treatise *Computational Techniques for Fluid Dynamics*, provides detailed solutions for the problems which appear in that book. The solutions are presented in enough detail for the reader to complete any intermediate steps. Many of the problems require computer programs to be written, and in some instances completely new programs have been provided here; their listing forms part of the solution. In addition, many of the problems are substantial enough to be considered mini-projects, and they should encourage the reader to explore extensions and investigate the problems further. Although targeted at instructors, the manual should be of considerable interest for practising engineers and scientists who need to acquire computational fluid dynamics skills.

Scientific Computation

ISSN 1434-8322

ISBN 3-540-54304-X

9 783540 543046

http://www.springer.de